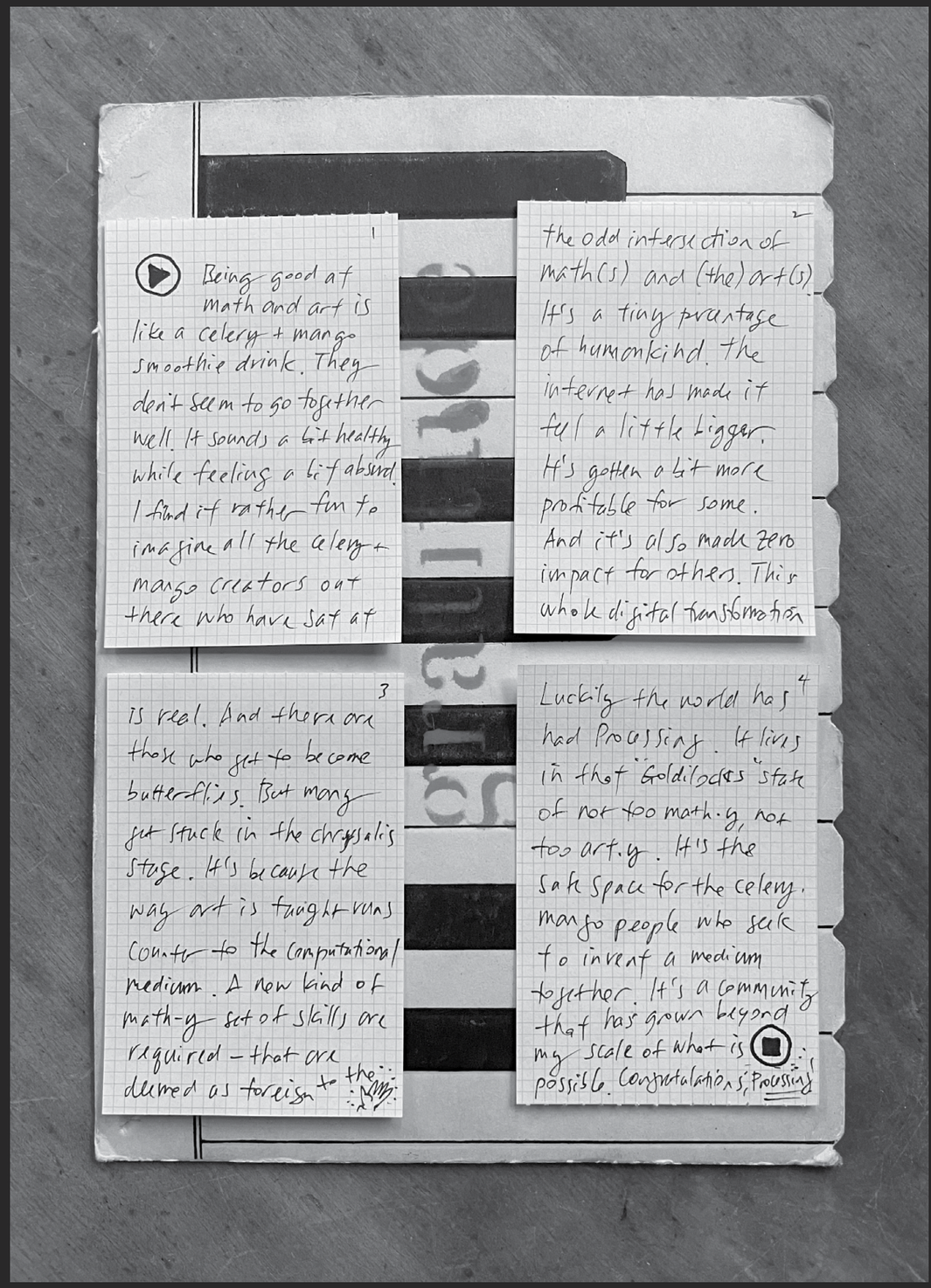
PROCESSING COMMUNITY CATALOG 2001–2021

# PROCESSING COMMUNITY CATALOG 2001–2021

The Processing Foundation


1
Being good at math and art is like a celery + mango smoothie drink. They don't seem to go together well. It sounds a bit healthy while feeling a bit absurd. I find it rather fun to imagine all the celery + mango creators out there who have sat at
2
the odd intersection of math(s) and (the) art(s). It's a tiny percentage of humankind. The internet has made it feel a little bigger. It's gotten a bit more profitable for some. And it's also made zero impact for others. This whole digital transformation
3
is real. And there are those who get to become butterflies. But many get stuck in the chrysalis stage. It's because the way art is taught runs counter to the computational medium. A new kind of math-y set of skills are required – that are deemed as foreign to the
4
Luckily the world has had Processing. It lives in that "Goldilocks" state of not too math-y, not too art-y. It's the safe space for the celery-mango people who seek to invent a medium together. It's a community that has grown beyond my scale of what is possible. Congratulations, Processing!

“Being good at math and art is like a celery + mango smoothie drink. They don’t seem to go together well. It sounds a bit healthy while feeling a bit absurd. I find it rather fun to imagine all the celery + mango creators out there who have sat at the odd intersection of math(s) and (the) art(s). It’s a tiny percentage of humankind. The internet has made it feel a little bigger. It’s getting a bit more profitable for some. And it’s also made zero impact for others. This whole digital transformation is real. And there are those who get to become butterflies. But many get stuck in the chrysalis stage. It’s because the way art is taught runs counter to the computational medium. A new kind of math-y set of skills are required—which are deemed as foreign. Luckily the world has had Processing. It lives in that “Goldilocks” state of not too math-y, not too art-y. It’s the soft space for the celery + mango people who seek to invent a medium together. It’s a community that has grown beyond my scale of what is possible. Congratulations, Processing!” John Maeda, 2021.

In August 2021, we celebrated the 20th anniversary of the first release of the Processing software, “Release 0001 — built 9 August 2001 in Japan, 6:01:26pm (5am Eastern).” The idea for the Processing Community Catalog which you are now reading emerged while planning this anniversary. We announced the catalog with this short note:

> What does being a part of the Processing community mean to you? We’re creating a community catalog to celebrate all we’ve done together with Processing for the last 20 years! We hope you will consider contributing a page that represents how you’ve been a part of the community. You can share art you’ve made, code you’ve written, an event you organized, a workshop you taught, or anything else! Your page can be anything—a sketch, some writing, a photo, a screenshot, a collage, a poster, a mini-zine, etc! ... We’re excited to see what you share, and please share this call with your friends.

In addition to the hundreds of pages created by the community and presented here, we’ve also included documentation of the activities of Processing, p5.js, and the Processing Foundation over the past 20 years. The community and work around these projects extend far beyond what could be included in one publication, but we hope this catalog offers a window into our collective, ongoing, and evolving process.

Here’s to 20 more years together!

Lauren, Casey, Qianqian, Ben, Cassie, Dan, Dorothy, evelyn, Johanna, Kate, Toni, and Saber

At our core is the philosophy and politics of FLOSS (Free, Libre, Open Source Software.) We see software as a medium, something that connects two things. We view it as a means for thinking and making. We believe it should be free. We believe that learning to program is not about acquiring a certain skillset, but is instead a creative and exploratory process. We believe software, and the tools to learn it, should be accessible to everyone. We believe software literacy and an understanding of media of all kinds is essential knowledge for today.

The history of Processing is many different things to different people. Casey Reas, Ben Fry, Daniel Shiffman, Johanna Hedva, Saber Khan, Lauren Lee McCarthy, Dorothy R. Santos, evelyn masso, Cassie Tarakajian, and Qianqian Ye reflected on what it meant to them in this conversation that was woven together and edited by Liz Stinson for AIGA Eye On Design in November 2021.

ORA 001

# An Oral History of Processing, 2021,

Liz Stinson.

Computers and design have long been intertwined. This relationship dates back to the 1960s, when computational pioneers used code to create novel forms. Since then, programming's grip on designers has only tightened, even as it's been obscured by the user-friendly interfaces of modern creativity software.

The bridge between technology, design, and art has always been in the exploration of tools, and how those tools enable code to become a medium for new forms of expression. In the late 1990s, this dynamic was at play at the Massachusetts Institute of Technology (MIT), in a lab called the Aesthetics + Computation Group (ACG). Started by John Maeda, the ACG was an update to and continuation of the Visible Language Workshop, a research group founded by Muriel Cooper (https://eyeondesign.aiga.org/muriel-coopers-visions-of-a-future/) in 1975 that explored how computation could be used in the field of graphic design.

Cooper was concerned with how computers could be leveraged to expand the notion of traditional publication design and its accompanying elements, like typography. When Maeda joined the Media Lab in 1996, he pushed at the edges of Cooper's work, exploring what code could enable formally, in the contexts of both art and design.

Cooper and Maeda established a long lineage of designers and artists who were interested in pushing the boundaries of what code could create. Among them were Ben Fry (https://benfry.com/) and Casey Reas (https://reas.com/), two research assistants in Maeda's group. During their time at MIT, Fry and Reas began to question how programming was taught to visually minded students. They wondered: How could they make programming more accessible to designers and artists? And what would it look like for code to become both a creative medium and part of the creative process itself?

In the early 2000s, Fry and Reas started building a piece of software that would let people code in a simplified environment using a variation of the Java language. The software, called Processing (https://processing.org/), was designed as a digital sketchbook where novice and experienced coders alike could create interactive graphics. Processing was more than a tool, though. It was a community—one that would eventually be built by thousands of people who have contributed code to the open-source environment over the last 20 years.

What started in 2001 as a side project has since become a worldwide environment used by thousands of people to explore the creative possibilities of coding. Here is the story of Processing, in the words of the people who created it.

Casey Reas, Co-Founder of Processing, artist (https://reas.com/), Professor in the Department of Design Media Arts at UCLA.

Ben Fry, Co-Founder of Processing, Founder of Fathom (https://fathom.info/) Design Studio.
Dan Shiffman, Associate Arts Professor at New York University's Interactive Telecommunications Program.

CR I don't think Processing would exist without John Maeda. The story starts there. Ben and I both came to MIT to study with John specifically because he was bridging ideas of computer science and together. At that time, he was the person who was making those connections, and that's what Ben and I were both interested in. We were both John's research assistants in the Aesthetics + Computation Group, where we were working with John on Design By Numbers (https://mitpress.mit.edu/books/design-numbers) (DBN), which was a programming language and environment and system to teach designers the basics of coding.

BF DBN was under-featured, and intentionally so. Like, can you have a programming language that's eight different commands and a limited screen of 100 x 100 pixels, and only grayscale? It was a super-minimal environment that was meant to just get people thinking about using computation at all.

CR I remember there were a few moments that were really pivotal around teaching DBN. One was at the AIGA headquarters [in 1999]. We did a workshop for designers—nontraditional students of all different experience levels. Ben, Elise Co (http://www.artcenter.edu/interaction/faculty/elise-co/), and I also taught a few-day workshop at RISD. Those workshops really opened my eyes. We could sit with a group of people who had never coded before—people who were designers—and within an hour, they were making stuff. But at the same time, there were also these questions of like, "Oh, I want to use color"; "Oh, I want it to be bigger than 100 x 100 pixels." And that was really the start of what Processing became.

BF Casey and I had kept making an experimental version of DBN that looked the same but underneath you could do OpenGL and 3D graphics and all these other, much more complicated things. There were secret commands to make it a larger drawing canvas or add color or switch over into using Python instead of the DBN language. We had all these odd variants that we would hack in as things to try out. But in the end, we didn't want to mess with DBN. There was something really wonderful about its simplicity, and there wasn't a good way to have DBN ladder up into something more complicated.

CR Processing tried to take the minimal aspects of DBN but also allow it to extend to the point where it was no longer purely a learning environment, but actually a full design and studio environment.



BF What we were after was: Can we take what is so nice about DBN where you write a little bit of code, hit "run," and something happens, and can we make something that's more sophisticated like the Java language? Can we wrap that up in a way that still feels as efficient and friendly and responsive?

CR In the early days, the Processing editor had the exact same user interface as DBN. I did the DBN user buttons and just moved them over to Processing. Over the years we've really simplified it, so now it's just really "run" and "stop," as opposed to like 50 buttons you can find on some of these "professional" programming environments.

DS Having the icon be basically the "play" button was a big innovation of the friendly language of Processing. It isn't "compile"—it's "play."

CR That was intentional. If you were learning computer programming at that time, you were doing classes in a computer science department. You were only working with text and math, and for visual people—people who are about the sensation of aesthetics—working in those classes, you would either stay for a few weeks and leave, or you would stay for a year and you'd be in pain.

BF I took some computer science classes during undergrad, and there was a friend of mine in the design department who was also taking one of the courses. I was so excited because he was super talented; this really brilliant person. And I'm like, great, I'll have somebody else who's in this class and we're gonna be able to kind of commiserate or whatever. We got a couple weeks in, and he dropped out of it.

DS That's a common story.

BF It's this sort of thing that's really burned in my memory. I remember bumping into him while I was on my way to class, and him being like, "Man, I just can't do it." And it killed me because we're learning all this other bullshit about data structures and algorithms and whatever, and it's so far from what he wants to do.

CR We saw a potential for learning how to code in a more essential and foundational way—that was really the vision of Processing. I don't really think that was happening at the time. Instead of, to say it in a really crude way, dumbing down coding so designers can understand it, we thought designers and artists and architects can really do a great job with this. It just needs to be presented to them in a way that is motivating for them and engaging. And the web was just extremely essential. The idea that you could export your work, throw it up on a server, and anybody who had a network connection anywhere in the world could see it. Putting the code people have made right there, so everybody can access it and

share and learn, was something we picked up from the browser, automatically allowing you to view HTML.

BF For me, learning how to code was entirely based on the willingness of other people to share their stuff. That was during a time before "open source" was a term that was coined, but there was just sort of a norm around it. If you were making stuff, you were probably going to share it. There was a willingness to do that. Later that started getting wrapped up in the open-source movement. There was a certain amount of "paying it forward" but also just going back to the idea of, how do you set a baseline for the community around the willingness to share? How do we try to get away from the obfuscation of things and closer to the explanation of them?

By 2001, Reas had graduated from MIT and was teaching, while Fry was working on a PhD. Both were developing Processing into a scalable coding environment on the side. What started as sketches in their notebooks became an amalgam of Java and new programming elements which people could deploy as "sketches," the term Fry and Reas used to describe a Processing program. The change in terminology was both symbolic and practical. They wanted Processing to feel accessible, while at the same time enabling experienced programmers to use the language as a way to break free from the typically rigid coding environment. This novel approach to coding caught the attention of Shiffman, a recent graduate of New York University, who was interested in bringing the Processing language to the school's Interactive Telecommunications Program (ITP).

CR The traditional way of writing computer code is to figure everything out and then to transfer that into source code. The idea we pulled from the arts was: You don't really often know where you're going. The only way that you figure out where you're going is to walk the walk, and to go down all the different paths.

BF In the visual arts, you can sit with your sketchbook and scribble things out. If you screw up or do the wrong thing, you can flip to the next page and start again. That's necessary because as you start working through ideas, a lot of them suck.

CR The goal was to make the creation process a little bit more informal. To get people to say, "I'm going to sit down for an hour, and I'm going to make five sketches, and then they're all there in my sketchbook."

BF The sketching aspect of it is essential to actually working out the idea because you can't, unless you're a genius, just completely conceive of everything in your head and have it come out and work all the time, right off the bat. We wanted this [software] to be able to give folks a way to do half a page of code and see what happens. Because if you wind up having all these other steps and you



have invested all this time into just getting something on the screen, that makes you work in a really rigid way.

DS Ben once said something about never joining more than two words in a function name. All these computer programs are so wordy—you know, "Java factory object builder dot like blah blah blah." So the idea of the sketching and the sketchbook is kind of like the foundation of all of that thoughtfulness around language.

BF I think it works well for people who are getting started, but even if you actually understand this stuff well, there's no reason to further burden yourself with all this extra nomenclature. We wanted to make sure that the things that should be simple or things that are going to be used a lot were really efficient and really clear.

DS I recall at the time that the two different tools I was using were Director and C++. I don't know that I ever really liked Director, although I appreciated it. The interface drove me crazy. There was a timeline and a stage and all that stuff. I just wanted to play with math algorithms and visualize them. Or I was working with C++, and that also drove me crazy because it was really difficult to debug. Processing just landed in this perfect sweet spot between the two. It was kind of a right place, right time thing in that I was graduating, and I proposed (or maybe Red [Burns] suggested it to Dan O'Sullivan, I don't remember) something like: I see Lingo being used, and I see C++ being used in this graduate program at NYU. I think Processing would be really great to take over. It's never been about a competition—it's a rising tide lifts all boats kind of thing—but being able to basically port the curriculum that used Director to Processing was one of the first things that I did [when I joined the faculty].

BF I remember around 2004, there was a transition from a lot of folks being like, "Oh yeah, Processing. I've heard of that; I might try it out." To like, "Oh, Processing. I actually used that for my thesis." And I was like: *Did you graduate? Is everything okay?* There was sort of an inflection point of it suddenly being in use, and we probably owed ITP a great deal for that.

CR When I was traveling, everywhere I would go, I would find all these people using Processing.

BF It just kind of grew very steadily from there. It's sort of funny because we have a chart of the users for growth, and it's this very straight line going steadily upward. Over the years there were various points in the history where we're like, "Okay, this is as big as this audience will get," and then it just kind of kept growing.

CR There were a number of institutions really early on who I would say took a chance, and it usually came from a student, or a young faculty member who wanted to do a workshop. This was way before there were classes around Processing, and

a lot of the software development happened through needing to get out a new release for a workshop. Sometimes Ben and I were doing a workshop together, and we'd release the software at 3 a.m., and then be in class at 9 a.m. to teach. It was largely pushed forward by necessity.

As Processing became one of the primary environments for creative coding in the early aughts, a community started developing around it. People were eager to use the tool, and a smaller subset of them wanted to help develop the infrastructure required to maintain the growing piece of software. Community members from around the world contributed to Processing Libraries (https://processing.org/reference/libraries/), which extended Processing's capabilities and expanded the language to work in different mediums like sound, hardware, and computer vision. The software itself continued to evolve, but the support system behind it was lagging. Since its inception, Processing had been a labor of love. By 2011, Reas and Fry decided to formalize the organization's work by establishing a foundation that could help raise funding and expand access to Processing through educational efforts.

Jonanna Hedva, Director of Advocacy at Processing Foundation.
Saber Khan, Education Community Director at Processing Foundation.
Lauren Lee McCarthy, p5.js Creator.
Dorothy R. Santos, Executive Director of Processing Foundation.

CR Processing had been around for about 11 years by the point we started thinking about the foundation. We had been working completely on a volunteer basis. We probably should have formed the foundation about six years before we did because we were always really struggling to get the software released, and we didn't have a lot of energy or ability to plan for the future. We had a few pieces of funding come through from different institutions at different points in time, but the project had really just grown to a scale that we needed to start working with money and started applying for grants. So the foundation really came out of necessity of needing to fund the software development.

JH I started working in Casey's garage in the summer of 2014. At the time we had the 501(c)(3) status, and that was sort of it. There was no clear plan or strategy or vision. In the early days it was really trying to figure out how to translate the 11 years of work with the software into a nonprofit foundation that would serve some sort of community. There was already a thriving community, so I think the question was: What role can the foundation serve to support what's already happening? And is there anything that we could do to catch the momentum and bring in more support?



CR An initial idea was, let's bring in money, and we'll be able to hire some developers—because the people who were developing the software at the time just had too much responsibility and too much weight on their shoulders to be maintaining such a huge piece of software for such a large international community. It just wasn't possible for them to carry that burden.

LLM We were at a point in 2015 or 2016 where we felt like everyone was working at capacity. There was this moment of: Okay, we either have to scale back what we're doing so it can be manageable, or we need to scale up a little bit to build some infrastructure so we can really think about sustainability. I remember asking, "Who wants to scale back what we're doing?" And nobody wanted to, so we had to go the opposite direction. It's really been about trying to think about what we need to support what we want to do in a way that is manageable and also creates opportunities for people who can't afford to volunteer their time to these projects.

JH Open source is tricky. It's kind of a paradox because the whole point of it is that anyone can use it, participate in it, and build it, which also means that there's no money. It's free for everyone to use. You can participate in the community, but it also requires a lot of volunteer labor, which of course excludes an entire swath of the population outright.

CH My one-liner is that free software is expensive to make. In reality, we weren't successful as a foundation in funding development in the early years, but the foundation really expanded the project in a lot of ways that I think were unforeseen by me.

JH It became very clear, very quickly, that things like diversity and inclusion and access were actually something that we could do as a nonprofit and foreground as a priority. We really wanted to try to support community members who hadn't been supported in other open-source places and contexts before that. The thing that felt really important to me about Processing was not necessarily what the potential is to create, although that feels really expansive. But the potential to change how these things are seen, what code can be, and what people who contribute to software projects could look like. The transgressive potential to me is almost more exciting than the thing of like, "Oh, we could make a cool animation," or something. I mean, that's nice too, but to me this is very political, actually.

DRS 100,000% to everything Johanna said. The way that one contributes to a project can be so multifaceted and can take on so many permutations. For a project or open-source software project to be able to do that and to kind of have that ripple

out into not just even like a local or U.S. community, but a global community—that's very hard to do.

LLM There's a way in which we're taught to believe that a technology or tools are just about the code. And I think that's totally false. A contributor is someone that contributes, not necessarily writing code.

SK I think all those things make Processing a really important voice in the K-12 education space. In K-12 education there's a lot of interest from Silicon Valley and corporations to have their curriculum and their presence [in schools]. I think this makes it really important that the software gets made by people who have politics and interests. I find that often in K-12 there's the sentiment of, you can have fun when you're a kid, but when you grow up it's gonna get really serious.

DRS I don't want students to ever feel that they must learn Processing or that they must learn programming in order to be successful in the world. But this is also a part of the challenge of getting funding. Oftentimes you have to prove the worst of a project in terms of its output—not because of what it teaches an individual.

CR I just believe in the value of the arts. We've always approached coding in the same way that it's useful to learn how to play a violin or a recorder when you're in elementary school—it enriches life. We're not training worker bees. We're allowing people to be creative and expressive in this emerging way.

DRS I actually took one of my first Processing workshops at Gray Area in 2011, and I remember I felt very embarrassed and ashamed because I literally thought to myself, "Everyone in this 101 beginners workshop knows how to code really well." They're making like freaking wormholes in their laptops, and I was excited to make a circle that could bounce up and down.

LLM One of the things we thought about really early on with p5 [note: more on p5 in just a moment] was the culture on GitHub, which is like the main place where the code base is developed. A lot of the conversations can be very aggressive, and you have to put yourself forward as an expert in order to be listened to. And we just really wanted to change that dynamic.

Around the same time that the Processing Foundation was established, another project started to take form. In 2013, Lauren Lee McCarthy ran into Reas at a conference and they began talking about the lack of diversity in the open-source coding space. Soon after, Reas reached out to McCarthy to see if she wanted to work on Processing. When McCarthy joined the project, she began thinking about how Processing's initial goals of access could be updated for 2013. She believed that a web-based coding environment could make the software more accessible to people, from both an educational and technological perspective. The result was p5.js, an update to the original Processing tool built for a web interface that used JavaScript as its core coding language. In 2019, McCarthy stepped down as the lead of p5.js after six years of leading the project in order to make room for a new generation of leadership. It was a moment of transition for the organization, as it searched for a way to make leading a large-scale open-source project more equitable and sustainable. p5 embraced a novel rotating leadership structure that would pass the reigns from leader to leader every year as a way to breathe new ideas and perspectives into the project. Today, p5 continues to focus its efforts on expanding its reach to people outside of the typical creative coding profile.

evelyn masso, p5.js Co-Lead.
Cassie Tarakajian, p5.js Online Editor Lead.
Qianqian Ye, p5.js Co-Lead.

LLM p5 really started during a conversation between Casey, Ben, Dan and me where they asked: "What would Processing look like if it were reinvented today?" And so the first thought we had was, well, it would live on the web again, and it would run in all the browsers really easily. Beyond that, we were also thinking about how we would take that early idea of expanding access to this field and kind of update that in 2013. What does access mean to us at this point, and how can we evolve that idea?

CR In 2001, Processing was completely entwined in the world wide web. You would export from Processing, you'd put it up on the web, and you'd share it with everybody. But because technology moves and changes so quickly, the ability deteriorated over the years. p5 allowed this way of thinking—a lightweight approach to entering into the web again. It was so essential.

BF What's so fun about stuff like p5.js is that it gets us so much closer to that idea of turning the computer on and Basic is just there. The browser is this sort of lingua franca thing that works everywhere. The idea of having folks be able to connect code right in that environment was super exciting.

LLM I started working on p5 within the context of an early fellowship, which is a program that the Processing Foundation runs. It was really just like 100 hours of work where I was experimenting with this idea. Pretty early on, I joined with Evelyn Eastmond (http://www.evelyneastmond.com/), who I was co-teaching with at RISD, and we kept developing this idea and teaching it to our students.

CR At the time, in order for high school and middle school students to work with Processing, they would need to install it on the computer labs in schools. A lot of the IT folks who were running these labs didn't want to install this piece of open-source software from an untrusted source. It created quite a bit of friction.

With p5 being able to code directly through the browser, that whole difficult administrative hurdle was taken out, and it became far more accessible.

SK In 2013, I was not aware of Processing at all. I started getting into teaching tech around 2012 or 2013, and like a lot of teachers, found myself needing to learn more. A lot of folks in New York suggested checking out Processing, but it wasn't what I was looking for. When p5 came out, that made a huge difference because a lot of logistical questions kind of got solved.

LLM One of the things I felt early on with p5 is that it wasn't just about having a diverse community of users; it was really about asking who's making the tool, who's contributing to it, because any tool is going to be embedded with the biases and the perspectives of the people making it. Some of the things we did really intentionally were saying, okay, let's expand what it means to contribute to a tool. It's not just writing core code, it could be making graphics, it could be doing outreach, it could be organizing, it could be teaching—these are all building this tool together.

CT Part of the impetus for the web editor was a lot of this access stuff. When I first started learning how to program, I remember day one of coding class my professor opened up his computer, opened the terminal, and I was just like, what is happening? No one explained it [coding] to me, and I really wanted to be part of a space where you could start writing code and it didn't feel like you were immediately doing something wrong.

LLM It's been this continuous process of trying to ask ourselves what access actually means, and then trying to address it in different ways. Less of a top-down strategy and much more of a bottom-up, constant learning where we realize when we got something wrong or missed something, and then think about how to address it.

EM I remember at the 2019 Contributor Conference (https://p5js.org/community/contributors-conference-2019.html) we talked a lot about how we make it more clear that the access work is kind of like the most important part of this.

LLM We've been thinking a lot about disability. Normally the elements that we're using under the hood to make graphics on the screen are as ones that are not readable to someone who is using a screen reader or someone who is blind or has low vision. We've been doing one project that's been led by Claire Kearney-Volpe, Luis Morales-Navarro, and Mathura Govindarajan, and they've been thinking about how we incorporate image descriptions into the code itself, and how we incorporate audio output so it crosses different domains.

EM There have definitely been more tutorials for more and more different kinds of people, especially in different languages but also different age ranges.



QY We have seen people using p5 in K-12 education, and there are more people creating work to make p5 accessible for elders. In 2020, two of the Processing Foundation fellows, Inhwa Yeom and Seonghyeon Kim, made a project called p5 For 50+ (https://p5for50.plus/about/) that created materials and tools to teach p5 to people who are older than 50. All of their teaching materials are in Korean because that's where they had all of their workshops.

CT I think having a voice in the tools that are being built is important. If you step back and think about what the point of technology is—it's to help us, right? And if the people designing the technology don't know who you are, they're probably not going to be able to help you. This was part of the foundation of Processing. What are tools that you can use to do art with computers? It's Adobe products, which are a black box that you pay for and you have no say over and then at any moment could just disappear. If you give people the reins, then they'll make themselves.

LLM We didn't start from zero, though. Processing had been made, and it wasn't just the syntax or the code that we were building off of—we were building off of the experiences that Ben, Casey, and Dan had, not to mention the Processing community and a lot of other creative coding and open-source communities. We were able to learn from that and get advice from those community members in making some of the decisions we did. One of my big hopes for this work in general is that these projects don't have to try to reinvent the wheel and have all the same problems. We should understand how much we're part of a larger Processing project, but we're also part of a much larger community, and these are all building on each other.

“How does one speak up to challenge the establishment? I think nuance and candor are essential to start a conversation. No one likes to be criticized, but everyone loves to be understood. When I was learning to code, a friend told me “treat your computer like your younger sibling.” This approach of care and play is essential in exploratory and poetic computing. I think a similar approach is necessary to navigate the tech community. As this project grows in it’s scale and diversity, I want to ask. Who do we make our oddkins?” Taeyoon Choi

PER 001 A Modern Prometheus, The History of Processing, 2018, Casey Reas and Ben Fry.

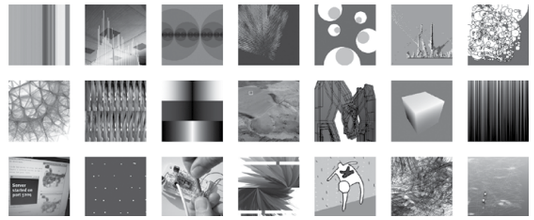


FIG 001-01 Selection of images created from Processing examples in 2003 to show a range of 2D/3D techniques, drawing with geometry and photography, and networking and physical computing (electronics) capabilities.

Grid of images, abstract graphics, illustrations, and photos.

When we started Processing in 2001, the goal was to bring ideas and technologies out of MIT and into the larger world. One idea was the synthesis of graphic design with computer science, combining the visual principles of design with ways of thinking about systems from computer science. We also wanted to share a way of working with code where things are figured out during the process of writing the software. We called this *sketching* with code. A third idea was to share what we had learned about how to teach programming to designers—to share this beyond the people we could teach directly through our workshops and classrooms. We wanted to spread this as far as we could.

This was all made possible by a set of programming tools we created specifically for making pictures, for choreographing animation, and for creating interactive work. Over many years, we refined a set of elements for creating visual design with code. Additionally, we didn't start this work from scratch, we built on top of existing ideas and code from people who worked in this area before us.


FIG 001-02 Information Landscapes. Muriel Cooper, David Small, Suguru Ishizaki, Earl Rennison, and Lisa Strausfeld, 1994. Image copyright Massachusetts Institute of Technology. Courtesy Visible Language Workshop Archive, MIT Program in Art, Culture and Technology (ACT).

Three-dimensional text at different scales, creating angles in space.

The Lab

Processing has direct origins at the MIT Media Lab going back to the Visual Language Workshop (VLW.) The VLW was founded in 1975 and it became a founding research group at the Lab from 1985 to 1994, when Muriel Cooper, the director, passed away. Cooper was a graphic designer at MIT, first at the Office of Publications and later as the founding art director at MIT Press. The VLW focused on traditional design considerations within the domain of screen-based publication and information spaces. One of the most

compelling and public demos to emerge from the VLW was *Information Landscapes*, a three-dimensional presentation of animated, interactive typography that was well ahead of its time. Processing was hugely influenced by the research and people within the VLW.

Processing emerged directly from the Aesthetics and Computation Group (ACG), a research group started at the Media Lab by John Maeda in 1996. Maeda's research at the Lab continued to synthesize visual design exploration with emerging software technologies. The students he recruited worked within similar themes to the VLW and their work continued research into programming tools to create visual media. David Small was the one person to start in the VLW and to graduate with his PhD in the ACG. His work and knowledge established a continuity. Within the ACG, code was most commonly written with the C++ programming language and a graphics toolkit called ACU that was started by Ben Fry, with Tom White and Jared Schiffman contributing significant portions as well. In the late 1990s, when we started at the Media Lab, ACU was used on Silicon Graphics Octane and O2 workstations—computers that cost tens of thousands of dollars at that time. By the year 2000, ACU was used on PCs configured with a graphics hardware that cost hundreds of dollars. This shift in hardware affordability is one of many reasons why, at the turn of the century, it became possible to create the kind of work created within the VLW and ACG outside of academic and corporate laboratories.

The ACG website remains online at http://acg.media.mit.edu/ and there are a few personal backups. A slice of the work is archived in Maeda's *Creative Code* book, published by Thames and Hudson in 2004. The ACG concept videos (http://acg.media.mit.edu/concepts/) that Maeda encouraged us to make are low-resolution videos by today's standards, but they show the work in motion as intended. Most of the ACG MS and PhD thesis documents (http://acg.media.mit.edu/projects/theses.html) are online as well.

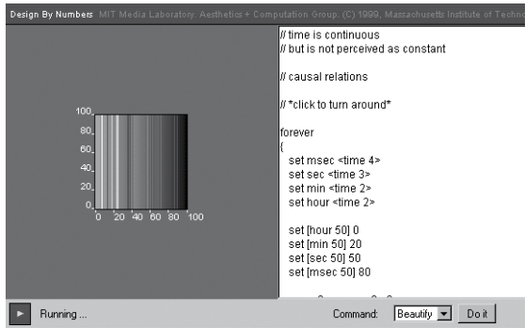



FIG 001–03 Design By Numbers online coding system within the Courseware, an online system for writing code, turning it in, and exhibiting the results. This screen capture was made in 2000.

Showing abstract depiction of time, code on right sets millisec, sec, min, hour.

John Maeda started the Design By Numbers (DBN) (http://dbn.media.mit.edu/) programming platform within the ACG. Both the MIT Press book and software for the project were released in 1999. Maeda brought the two of us into the project after the initial release to help maintain and



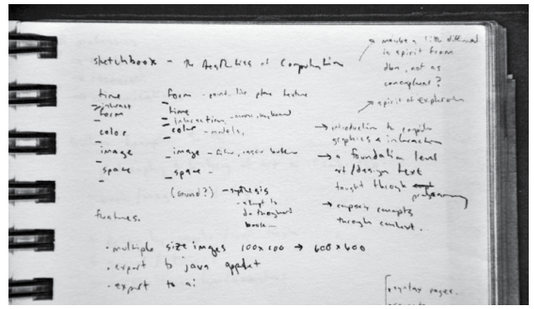



FIG 001–04 Notes in Casey's sketchbook in 2001 from our first conversation about the project that would become Processing. There was an early focus on creating a coding environment that would be compatible with the tradition of foundation studies in art and design education.

Notebook, handwritten text reads sketchbook—the aesthetics of computation, time, form, color, image, space.

extend it. Many aspects of Processing were modeled after DBN, which also integrated a code editor with a language. DBN was a minimal system, the canvas was always 100 × 100 pixels and only gray values could be used—there was no color. These constraints, as well as comfortable code elements such as paper and pen made DBN easy to learn.

Teaching with Design By Numbers was a formative experience for us. Our first event was a workshop at the AIGA national headquarters on August 25, 1999. This "computational illustration" workshop was planned for people with no prior background in programming and even for people without much experience operating a computer. Maeda led the workshop and Tom White also assisted. Maeda later asked us and Elise Co to teach a series of workshops at RISD. For these and other workshops at the time, we designed and built the DBN Courseware software, a series of Perl scripts that allowed teachers to create exercises and students to upload their work directly to a web server.

This experience kindled the ambition to start Processing. We started by extending DBN to include color and other features, but soon realized that these limitations were the essence of that platform and it shouldn't be expanded. We wanted to make a system that was as easy to use as Design By Numbers, but with a bridge to making more ambitious work. We wanted to allow people to work in color, at large sizes, to create 3D graphics, and more. Simple Processing sketches are almost as simple as DBN sketches, but Processing scales up—it has a "low floor" and a "high ceiling." The ceiling is more similar to the C++ programs we used to write with ACU partly because the rest of the Java programming language and its libraries were available.

Language, Environment, Community

We created Processing as three parts: language, environment, community. Each part is essential; the project is defined by the interactions between all three.

Language

A Processing program is called a *sketch*. This is more than a change in nomenclature, it's a different approach to coding. The more traditional method is to resolve the entire plan for the software before the first line of code is written. This approach can

work well for well-defined domains, but when the goal is exploration and invention, it prematurely cuts off possible outcomes. Through sketching with code, unexpected paths are discovered and followed. Unique outcomes often emerge through the process.

Processing isn't a language created from scratch, it's a hybrid between our own elements and the Java programming language. As a minimal example, this is how the standard "Hello World!" program is written in Java:

```
public class HelloWorld {
  public static void main(String[] args) {
    // Prints "Hello, World" to the terminal window.
  System.out.println("Hello, World");
  }
}
```

This program encloses the line that writes the text to the screen within two layers of additional detail that are important for large programs, but are confusing for a simple program. This is how the same result is achieved in Processing:

```
print("Hello World!");
```

This "Hello World!" example, however, has very little to do with the essence of Processing—writing code to make pictures. This is a more common first Processing sketch:

```
line(10, 20, 90, 80);
```

This code draws a line to the screen from coordinate (10, 20) to (90, 80). A more interesting short Processing sketch draws the line from the center of a 500 × 500 pixel canvas to the position of the cursor:

```
void setup() {
  size(500, 500);
}
void draw() {
  line(width/2, height/2, mouseX, mouseY);
}
```

Because Processing is made for creating pictures, the language includes elements specifically for working with form, color, geometry, images, etc. At the same time, any code that can be used in Java can also be used in Processing. The main idea is to make it easy to do simple visual things, but to also allow a more experienced programmer to do complicated things within the same language.

Environment

The *environment* is the software application that sketches are written within. The Processing environment is called the PDE, the Processing Development Environment. The current PDE for Processing 3 has just a few buttons to Run and Stop sketches and



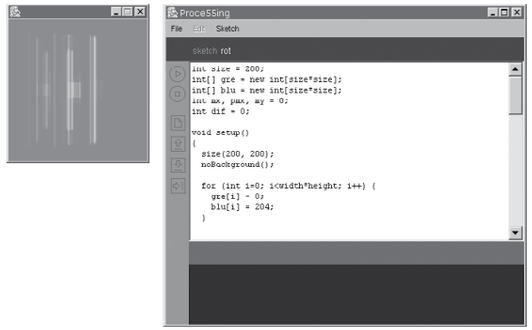


FIG 001-05 Processing ALPHA IDE running on Windows. Processing IDE, sketch with abstract white lines.

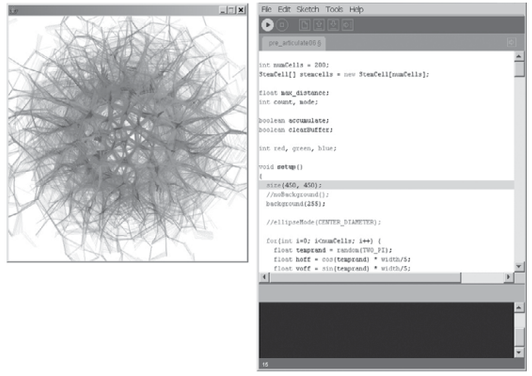


FIG 001-06 Processing BETA IDE running on Windows. Processing IDE, sketch with expressive lines bending and multiplying out from center.

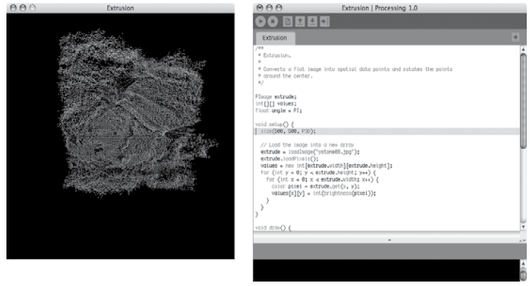


FIG 001-07 Processing 1.0 IDE running on Mac OS. Processing IDE, sketch with white dots making 3D form against black background.

to turn on and off the Debug mode. (The earlier PDEs had a few more to Open, Create, Save, and Export sketches.) The primary idea of the PDE is to make it quick and easy to start writing sketches. The secondary idea is to use the PDE as a sketchbook, a place to save the sketches and to provide easy access to open and run them. The PDE can also open and run example sketches and can easily link to the Reference.

The PDE was created for beginners and not everyone uses the PDE for writing sketches. Some people use other programming environments like Eclipse and other text editors like Sublime Text 2 to work with Processing. The PDE has evolved since 2001 with the Processing 3 PDE presented a major step forward.

Community

The *community* is the group of people who write Processing sketches and who share work and code with each other. We have never written custom software to support community engagement, but we have used existing Forum software and different community members have created new opportunities. Early versions of Processing exported sketches to Java Applets and at the time, that was an easy and convenient way to share work through the web. By default, exported sketches included a link to their source code and the web page included the text "Built with Processing." This phrase was a simple search term to explore work created by others and the source code included on the page was a way to learn how that sketch was created. Users could remove the links to the

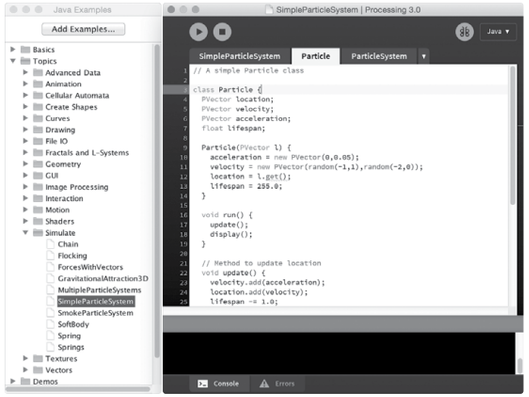



FIG 001–08 Processing 3.0 IDE with Examples menu open on Mac OS.

Processing IDE, examples menu shows many folders including Data, Animation, Drawing, Curves, Color, File IO.

source code, but we wanted sharing to be the default. We learned coding in large part by looking at others' code, and wanted this openness to be central to the project.

The first Processing Forum, at that time called *Discourse*, was launched August 2, 2002. The first posts in the *Introductions* sections were made by REAS, adrien, re|form|at, eviltyler, tomek, ik0, fdb, Josh Nimoy, Mike Davis, Takachin, edwardgeorge, riboflavin, jes, and Alex. For the first few years, the Discourse was a vibrant space. It was a place where people shared and helped each other. We were all exploring together and some people who knew more than others were generous in offering advice. The "alpha" forum was closed in 2005 and replaced by the "beta" forum, which was closed in 2010 for the "1.0" Forum. The complete set of prior forums are archived online: Alpha Forum (https://forum.processing.org/alpha/), Beta Forum (https://forum.processing.org/beta/), 1.0 Forum (https://forum.processing.org/one/), 2.0 Forum (https://forum.processing.org/two/). In May 2018, we launched our fifth forum (https://discourse.processing.org/), once again called Discourse.

In the spirit of community, individuals have created other opportunities for learning and sharing. The longest-running and most prominent effort is Sinan Ascioglu's OpenProcessing, which recently launched a new interface which is compatible with p5.js (https://p5js.org/) sketches. Earlier initiatives include the Free Art Bureau's *Processing Cities* initiatives to start user groups in cities around the world, Tom Carden and Karsten Schmidt's Processing Hacks wiki, and Tom Carden's blog aggregator. Early social media sites created community and energy around Processing through tags used within sites like Del.ici.ous and Flickr. OpenProcessing is going strong, but these other initiatives have changed as the web and the community have shifted.

The Processing source code has been available online for many years, first through SourceForge and later through Google Code, but our move to GitHub in 2013 started a new kind of community around Processing through increased quantity and quality of code contributions. GitHub makes it easier to integrate contributions and to discuss details. It also helped the expanded development teams to communicate and track software changes. A full list of contributions (https://github.com/processing/processing/graphs/contributors) to the Processing core software reveals this work. The most essential community contributions to Processing are the Libraries. There are over one hundred Processing Libraries (https://processing.org/reference/libraries/) that extend the software in different directions beyond the core. In categories ranging from Data to Simulation to Video & Vision, the Libraries are independent pieces of software that integrate into the Processing language. Most Libraries are developed by independent community members and the source code and examples are made available for everyone to use and learn from.

On Growth and Form

Processing started with a few notes in a sketchbook in spring 2001. In fall of that year, Ben taught the first workshop at Musashino Art University in Japan. This pre-alpha version of the software, Processing 0005, was minimal, but this early version of the language was integrated into a spare Processing environment. Because Casey was teaching at the Interaction Design Institute Ivrea in Italy at the time, many early workshops were in Europe. Processing evolved rapidly through these early workshops and the language was frequently changed during the alpha release period. We're grateful for the institutions that gave us a chance and the opportunity to spend time with their students.

In the early days, the project was called Proce55ing and the website was www.proce55ing.net. In the very beginning the project name was always changing: Pr0-c3ssing, Pro35sing, Pr0cess1ng, etc. We changed the name to "Processing" and the website to www.processing.org in 2004 to reach a wider audience. In general, the name of the project has received continuous snarky comments from people who don't like it, to frustrated people who can't find related information in search queries. The idea of the name "Processing" was to focus on "process" over final results and to indicate the active state of software; it's always running, it's realtime media.

For the first few years, Processing was distributed to people who signed up through the website. At that time, we were still working our way through the software agreements with MIT and the software was rough. The first www.proce55ing.net site went online Oct 20, 2001 with a set of examples, a reference, and the following text:

> *Proce55ing is an environment for programming images, movement, and interaction. It is a sketchbook for developing ideas, a tool for creating prototypes, and a context for learning the fundamentals of computer programming. The Pr0cess1ng environment is in its early stages and will continue to develop.*
>
> *Pr()ce55ing is written in Java and enables the creation of Java Applications and Applets within a carefully designed set of constraints. It uses a 2D/3D Java rendering API that is a cross between postscript-style imaging in 2D and 3D rendering with OpenGL.*

Before the 1.0 version of the software, we named the releases in order, rather than using the more traditional software release numbering (1.0, 2.0, etc.). For instance, Processing 0069 was the 69th release of the software. After we switched the style of numbering, we continued both for a time; Processing 1.5.1 is also release 0197. Revision 0069 was the last alpha release, revision 00161 was the last beta release, and release 1.0 is also 0162. The dates and notes for every release of Processing are online at GitHub (https://github.com/processing/processing/commits/master/build/shared/revisions.txt).

In 2007, we published the Processing textbook, *Processing: A Programming Handbook for Visual Designers and Artists*. This textbook, published by MIT Press, defined our vision for how Processing could be used in a university classroom. It was developed through years of direct classroom experience teaching at UCLA and Carnegie Mellon. The software is discussed within the context of the history of experimentation within the visual arts and technology. It includes interviews with professionals in the expanded fields of animation, performance, and graphic design. A set of Extension chapters expand the domain into audio, computer vision, electronics and other topics. This book, in addition to *Learning Processing* by Daniel Shiffman and *Processing: Creative Coding and Computational Art* by Ira Greenberg, were the first round of Processing publications that further extended the reach of the software in academia. A more complete list of books (https://processing.org/books/) written with Processing is published on the Processing website. Over time, energy has shifted away from books and onto online instruction and video tutorials.


FIG 001–09 Number of times the Processing software is opened on unique computers each month from 2005 to early 2018. The peaks and valleys are correlated with the academic year with the highest points in the fall and the lowest during the summer. This data doesn't account for shared computer use or when people turn this reporting off in the software Preferences.

Graph with exponential growth, cyclical valleys at the end of each year.

The number of people using our software continues to increase. As a part of this transition, the direct link between us and the original community spread so thin that the single Processing community has become a series of more fragmented groups associated around school, cities, social media platforms, or topics. The Processing software and the people who used it were largely synonymous in the beginning, but over time, it's become a piece of software that people use without a knowledge of the original context. This was a difficult and slow transition, but it provided new



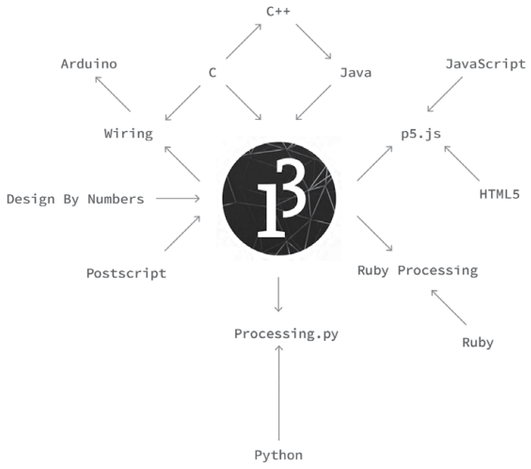



FIG 001–10 Processing has been influenced by other coding systems and it has influenced others. Processing logo in center, surrounded by graph of libraries including DBN, Wiring, p5.js, Processing.py, Postscript.

opportunities for more people to be involved and for the software and its core ideas to spread. As the community grew, it became increasingly difficult to determine how and why people were using the software, and therefore it became harder to focus on the development. Recently, the software has started to be used more frequently in high schools, mostly for classes in math and science. Universities continue to integrate the software into curricula; it has expanded into computer science and humanities departments.

The Present

We have had growing pains where the expectations of the community and the complexity of the software have outpaced our ability to maintain and improve it. The vast majority of the code is written by the same small number of people volunteering their time—there are no paid full-time developers. Technology shifts like moving from 32-bit to 64-bit operating systems and the emergence of high-resolution screens have created the need for significant updates even to continue running on ever-evolving operating systems and hardware. We started the Processing Foundation in 2012 along with Daniel Shiffman to expand our reach and to support the software development. Lauren McCarthy is now the fourth member of the Foundation Board and we have additional collaborators and advisors.

The original mission of Processing was to create software that made learning to code accessible for visual people (designers, artists, architects) and to help a more technical audience work fluidly with graphics. We aspired to empower people to become software literate—to learn to *read* and write software. We wanted to change curriculum in universities and art schools around the world. Instead of teaching students how to *use* software, we thought it was just as important to teach them how to *create* software.

The further expedition of the Processing Foundation is to make code more accessible to an even wider audience. Toward this goal, we're investing our resources in mentoring and in collaboration. We believe in the synthesis of the arts and technology, and we know the arts are a necessary part of education from a young age. We

do not want to live in a world where technology is developed without ideas and input from the arts and where only some people have access to learning to code.

Right now, we're at the start of Summer of Code 2018. Through this program, sponsored by Google, we're mentoring fifteen students to learn about software development. These students are working on different parts of the Processing Foundation software. We're also in the middle of our 2018 Fellowships (https://processingfoundation.org/fellowships) cycle. A Processing Fellowship is a grant with mentorship that allows an individual or group to realize a project within the mission of the Foundation. We awarded nine Fellowships this year. We're working with the Fellows on a wide range of projects: developing high school curricula, teaching coding on smart phones in Ghana, documentation for p5.js developers, improving the accessibility of our tools, and creating a new website build system for Processing.org, to name a few.

From the original Processing software, the Foundation is now supporting a range of different projects. The p5.js project is a JavaScript reimagining of Processing within the context of contemporary web browsers. This project was started and is led by Lauren McCarthy. Processing.py was started by Jonathan Feinberg and it's now a Mode for the Processing 3 editor. Additionally, Andres Colubri is extending Processing for Android as a Mode for Processing 3. Gottfried Haider has written code to get Processing 3 running well on Raspberry Pi and CHIP hardware, and he has written a library to read and write directly to the I/O pins. These inexpensive computers are in line with the original mission of Processing, to make learning to code accessible and enjoyable.

A more complete list of people who are contributing to Processing, and who have contributed in the past, is available as the People (https://processing.org/people/) page on the Processing website. Contributions through GitHub are diagrammed within the Processing Foundation repositories (https://github.com/processing).

We've been working on Processing now for nearly seventeen years. It's difficult to remember clearly what it felt like in 2001, when the first website went public or the day the Processing Alpha software was released on February 2, 2004. We have an archive of documents and "to do" lists and every change made to the code has been tracked, but this data doesn't capture the mood or personal impact. We hope this short text helps to bridge Processing in 2018 with its past. Most essentially, Processing is about people. It's about individuals and collective learning and exploration; it's about sharing ideas and giving what you can.



## PER 002 createCanvas: Saber Khan interviews Daniel Shiffman, 2019, Saber Khan and Daniel Shiffman.

FIG 002-01 Artwork by Alon Chitayat (https://www.animishmish.com/).
Daniel Shiffman in shirt and tie gesturing in front of a rainbow, unicorns, floating cupcakes.

createCanvas (https://soundcloud.com/processingfoundation) is Processing Foundation's new education podcast, which focuses on teaching at the intersection of art, science, and technology. The podcast is part of our new Education Portal (https://processingfoundation.org/education), a collection of free education materials that can be used to teach our software in a variety of classroom settings. Rather than endorse a specific curriculum, we've engaged with a variety of educators from our community, ranging from K-12 teachers, to folks who lead workshops at hackerspaces, to university professors in interdisciplinary departments. We've asked them to share their teaching materials, which anyone can use.

createCanvas kicks off with an in-depth, two-part interview with Dan Shiffman! Dan is the beloved host of The Coding Train, the vibrant YouTube channel of weekly creative coding tutorials. Dan has been part of the Processing Foundation since before it was a foundation. In this episode, he talks to Education Community Director Saber Khan about how and why he started making educational materials for creative coding, what open source contribution can look like (spoiler: almost anything!), and he takes us behind the scenes of his YouTube channel, which currently has 889,000 subscribers.

Dan Shiffman, Associate Arts Professor at New York University's Interactive Telecommunications Program.

Saber Khan, Education Community Director at Processing Foundation.

SK Hi everyone. Welcome to createCanvas, a podcast about the Processing education community. I'm your host, Saber Khan, the Education Community Director of Processing Foundation. [Intro is the same as above.]

I'm here today with Dan Shiffman. Dan, do you want to introduce yourself?

DS Oh boy. Okay. Yes, I definitely want to introduce myself. My name is Dan, and I've been part of the Processing Foundation since the start. The Processing project itself dates back 10-plus years before we started the foundation. I started working with Processing in 2003, a couple of years after Casey Reas and Ben Fry started it. I started working with Processing by discovering it here where

we are sitting right now at ITP, which is a two-year master's program at Tisch School of the Arts, NYU. I haven't looked back since, and I've done a lot of things with Processing, teaching and making video tutorials and all that stuff, which I'm sure we're going to get into.

SK What do you do here at ITP?

DS My official, fancy-sounding title is Associate Arts Professor, and most of the courses that I teach are—the way I think of them is they're computer science-like classes. I look at a lot of topics that you would find in a computer science course, but through the lens of art and creativity and open source.

SK How did you get involved? You said you discovered Processing when you came to the ITP.

DS Yeah, so there were a couple people teaching with it, and doing workshops here. Amit Pitaru was teaching a class called Code and Me, and JT Nimoy had done a workshop (http://cdn.jtn.im/itp/p5/workshop/) that introduced Processing to ITP. The curriculum here, the learning to code curriculum was, at the time, using primarily Macromedia director.

SK Wow.

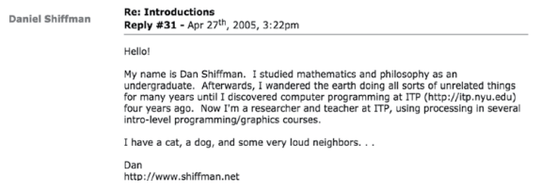
DS And the lingo programming language. I took a course in Java, the actual Java programming language. I took a course from Danny Rozin about doing computer graphics and image processing with C++ and Windows at the time. It was right around when I was graduating that I discovered Processing through these other workshops that were happening at ITP. I thought, "Oh, this could be a platform to teach our Intro to Programming Class." I was then hired to be a research resident. It was kind of a third year of being here and doing workshops and helping students. We still have research residents at ITP today that do this work.

So I did some workshops with Processing and then ran a little pilot test. It was called, Procedural Painting (https://shiffman.github.io/Procedural-Painting/). It was, basically, learn to code with Processing, and doing graphics and animation. So I did that course, wrote a whole lot of tutorials and handouts for it that I put online on our website. And then once I did that, it seemed to go well. Then we incorporated it the next year into our Intro to Programming class, and I kept iterating on those online tutorials, and those actually eventually became what is now the Learning Processing book (http://learningprocessing.com/).

SK It sounds like your involvement in Processing [happened] by sort of taking it on, and in creating a lot of materials, and making it accessible in the classroom.

DS Yeah. Who's to say what's an official contribution to Processing and what's just a





Daniel Shiffman

**Re: Introductions**
**Reply #31** - Apr 27th, 2005, 3:22pm

Hello!

My name is Dan Shiffman. I studied mathematics and philosophy as an undergraduate. Afterwards, I wandered the earth doing all sorts of unrelated things for many years until I discovered computer programming at ITP (http://itp.nyu.edu) four years ago. Now I'm a researcher and teacher at ITP, using processing in several intro-level programming/graphics courses.

I have a cat, a dog, and some very loud neighbors. . .

Dan
http://www.shiffman.net

FIG 002–02 Dan Shiffman's first post to the Processing forum in 2005. "Hello! My name is Dan Shiffman. I studied mathematics and philosophy as an undergraduate. Afterwards, I wandered the earth doing all sorts of unrelated things for many years until I discovered computer programming at ITP four years ago. Now I'm a researcher and teacher at ITP, using Processing in several intro-level programming/graphics courses. I have a cat, a dog, and some very loud neighbors... Dan."
Screenshot of email intro from Dan.

contribution to Processing. At that time, I was just any person on the internet who was interested and trying to get involved with the community. For a presentation that we gave at Processing Community Day, I found my first post to the Processing forum, which is like, "Hello, my name is Dan, and I'd like to participate in Processing."

There were no books at the time. I don't know that there were video tutorials, and there were just a few online tutorials and materials that were part of the Processing website. I was able to carve out a little niche there, of, "Oh, I'm writing beginner tutorials with Processing," and they were just flat HTML pages. Most of it was me being panicked and anxious about teaching. Teaching the class one or two or three times, I ended up accumulating this material, and over the years, other people started using that material, and I got in contact with Casey and Ben, and became more involved with the project as an open source contributor.

SK There's a lot of overlap between the Processing community, the NYU ITP community, and a few other ones. Do you mind explaining contributions that are sort of unofficial? How do you talk about the solutions?

DS One thing that I've been really lucky to have, that not all open source contributors have, is institutional support. That support is financial, and it's emotional, and it's infrastructure. At the time [when I first got involved], I was just a research fellow, a research resident at ITP. In that sense, my role was to help offer students support, and I was also doing that while [I was] teaching myself Processing.

Now, though, if you fast forward 15-plus years later at NYU, the job that I have here has multiple components. There's teaching, and I use Processing and p5.js for a lot of that. There's service to the university, which is running student clubs, or participating on committees, and all sorts of other kinds of things. And then there's research, or professional practice. If you're going to be at an arts school, that component is less traditional in the academic sense. It's not, "Oh, you must publish all these papers in these journals." A lot of people that I work with are practicing artists, or do a lot of other kinds of things. I have always made the case that the open source work that I do on Processing,

p5.js, and with the Processing Foundation, is both my research and my professional practice.

That's one way that helps allow me to do this work in a way that's sustainable. It has also helped Processing Foundation and Processing projects over the years. [Since] a lot of us who work at institutions have mechanisms for supporting this kind of work, we're able to basically do it without additional pay. So, and I'm cautious about saying this, I don't know that what I do is volunteer work for the Foundation. Maybe some of it is, maybe it isn't, but I do consider it to be the research and professional practice component of my job at NYU.

That's something that I feel is really important to be conscientious about in open source—because there are other people who are contributing to Processing, who are maybe freelancers or independent artists, and they're doing the same kind of work, but don't have that institutional support—and how can we make sure that [those] people can contribute. I want to make sure that being able to contribute is accessible to everyone—people who need childcare, or people who have different kinds of abilities, can also participate.

SK That's a really interesting discussion happening. But before we jump there, p5.js also has a relationship with ITP as well.

DS Yes. One of the things that I always try to do if I can, is funnel any types of funding or hiring opportunities from NYU towards the Processing Foundation. For example, in addition to the research residents who are recently graduated students, we have a program [for people] in residence. It's a research fellow-like program, so independent artists or teachers, or somebody on sabbatical from another job for a semester, can come and do a residency at ITP for a semester.

One of the people who has been doing that over the last several semesters, and continuing this fall, is Cassie Tarakajian (https://cassietarakajian.com/). Cassie is the lead developer of the p5.js web editor. This is a project that requires a ton of time and effort and expertise; a lot of that is technical expertise, a lot of that is community management. There're so many aspects of the web editor project. It really requires funding to keep that project afloat.

The p5.js web editor is an online platform for creating, editing, and sharing p5.js sketches. Cassie Tarakajian is the lead developer of the web editor.

One of the ways we've been able to fund that project, among others, is to have a residency for Cassie at ITP, where her primary work is developing the web editor. I can make the case for that because the web editor is the primary



platform for all of our introductory programming curriculum. This is a way that works: for an institution to support an open source project that's free for everyone, but isn't free to make.

We need to find clever ways to fund those [projects] beyond just donations on the Processing Foundation website. This is one of them. And that's not unique to ITP or NYU. Golan Levin (http://www.flong.com/) and the Studio of Creative Inquiry (http://studioforcreativeinquiry.org/) at Carnegie Mellon have provided lots of similar kinds of support. UCLA [has too], where both Casey Reas and Lauren Lee McCarthy are full-time faculty. University of Denver and Chris Coleman (http://www.digitalcoleman.com/) have provided a lot of support.

Then there're companies like Ben Fry's company, Fathom Information Design (https://fathom.info/). So this is something: having institutional partners. They might be writing a check, or they might be providing a residency, or they might just be providing free space. That's something that's been crucial and fundamental to how the Processing Foundation has been able to sustain itself. I say this because it's something other people who work at institutions could think about doing. Providing residencies, or making sure that, when faculty are being evaluated, that if they put their open source contributions, those are equal to publishing in closed academic journal–type things.

SK Yeah, and so much of the work that has come out of these places with Processing Foundation has now found its way into schools, to high schools and middle schools. [It may be] totally unexpected, but it can happen through open source work. It's hard to see that happen through writing papers, let's say, or something else.

DS Right, right.

SK It's had a very large effect.

DS Yeah. And I don't mean to denigrate the writing and publishing of papers. That work is very hard and important and meaningful, too. I just view it more as my role to advocate for open source contributions as professional research.

SK It's a known quantity versus "this is hard to capture."

We should transition. It seems like a continuation in your journey, but a very interesting and new area: You started working on the Coding Train. Do you want to tell us what that is and how you got started?

DS Yes. Back to your original question about how these pieces fit together, this is one that I often struggle with the most, [trying] to figure out where this fits exactly. I have my work with the Processing Foundation, and my work at NYU,

which is a very specific job. What grew out of these [is] this separate entity, this YouTube channel.

The origin story is that, at some point, I started making supplemental videos to go along with teaching my classes here at ITP. Those were in the form of screencasts, or in the form of trying to set up a camera in class. And none of that really worked effectively. I think both of those things were very useful. If I have a class of 16 students, recording the class was really useful for them, because they could go back and look at it. Or doing a screencast that was augmenting one aspect of discussion in class, that was super useful too.

But recording a class was problematic for bringing other viewers in. Number one is, I was always very concerned about privacy. I was like, "If a student is asking me a question, should I turn the camera off?" It also inhibited people. The classroom, whether it's three students, or 16, or 50, or whatever the number, is a sacred space, in a way. Audio recording, video recording, other than for that community's documentation, never felt right to me. So I started experimenting with trying to make similar videos that were more engaging and were produced outside of class, where I could be in them, talking and engaging with whoever the viewer might be, but not the screencast and not in class.

I started makeshift, setting up some cameras in my office, and I painted a wall green, and I would try to composite a green screen with computer captures. I was making videos that way, uploading them to Vimeo. They were supplemental materials from the class, and they were also a public resource. I was doing that for about a year, and then two things happened, which really changed the way I was doing that process. One was, somebody emailed me saying they wanted to watch the videos at 2X speed, which was not a feature of Vimeo at the time, only on YouTube. I had a hundred videos that I then uploaded all at once to YouTube. And two, I started becoming interested in experimenting with livestreaming.

The system I was using to record videos was using software called Wirecast, although now I use an open source software called Open Broadcast Studio. Wirecast is commercial software for doing the same thing, for taking a bunch of different inputs. I needed a video that was on me, and I needed to record the screen; I had the audio from the computer, I had my audio; and then I wanted it to be in a green screen so I could composite, but I didn't want to do any post-production. So I used software, Wirecast, or now Open Broadcast Studio, which is similar to what you would use if you were doing a sports broadcast, for example. It's much simpler [if you need] 10 different cameras to broadcast live and switch between them. I wasn't broadcasting live, I was just recording everything to disk. But then I realized, "Oh, I could broadcast this stuff live."

Once I started doing that, two things happened. One is YouTube, for better or worse, was a platform that reached a much larger, more international audience. People would find the videos in ways they wouldn't on Vimeo. So the audience started to grow that way. Then livestreaming really created this community. It's not the same as sitting in a class at a university, one teacher, however many students. It created this sense of knowing each other, or knowing me, and me knowing the audience in a different way. And so with both of those things together–YouTube as a platform and livestreaming–the audience started to expand and grow very quickly at a certain point.

It [became] something of an obsessive addiction for me to keep doing it. Some of that is probably the things that I associate negatively with the internet: that I'm just optimizing for clicks and views, and you get kind of addicted to that.

SK [laughs] Yes.

DS But also it was, "Oh, wow, there's a 13-year-old living in a country that I have heard of but know nothing about, who suddenly learned to code and made this whole project that they shared with me." Those types of moments are really thrilling and exciting to be able to reach audiences in a totally different way.

Now, it's become a little bit of a small business. It generates some income; I spend one day a week doing it. I started hiring other people to work with: a video editor, now a community manager. On the one hand, it's an extension of my work at ITP and with the Processing Foundation, because the material is the same material that I'm using to teach. I'm also using Processing Foundation software and packages in it. But it's kind of its own thing as well, in terms of being an entity that produces educational content on YouTube.

SK One thing I really like about the livestream is watching you think through a problem. How do you pick a topic to cover? What kind of research do you have to do? Is it like a cooking show where you have a cake that's ready? I mean, I know you're live-coding it, but have you thought through where you're going to go, those kinds of things? Do you have one example in mind about how you think through something like that?

DS I really struggle with this also, because, on the one hand, the thing that I think people have enjoyed or related to the most in what I do in these livestreams, is the working through it and getting stuck, and sometimes not being able to solve the problem. That is a thing that you do see more now. But when I was starting making video tutorials, more of what you would find would be well-produced,


FIG 002–03 createCanvas returns with Part 2 of our in-depth interview with Dan Shiffman! Dan is the beloved host of the Coding Train, the vibrant YouTube channel of weekly creative coding tutorials. Dan has been part of the Processing Foundation since before it was a foundation. In Part 2 of the interview, he talks to Education Community Director Saber Khan about different sustainability models for open source, the pros and cons of using YouTube as a platform, and how the Coding Train is more about community and documentation than it is about technical expertise.

Daniel Shiffman holding a ukelele and rainbow drawing in front of coding train illustration.

thought-through, curated lessons. [Now, it's that] people are able to see somebody else who they think (whether or not it's true) knows what they're doing, getting stuck and not knowing what they are doing.

That, I think, is empowering and helps people build confidence or realize that they're not alone in struggling. At the same time, if you're searching around the internet, and you want to learn how to load weather data from an API, and you find a three-hour video of someone trying to figure it out, maybe that's less useful in the moment as a five-minute video that shows the steps. I have not figured out how to strike this balance. But I'm trying to do a little bit of both.

Picking topics for a while was easy for me, because I had 10 years of teaching material, and I'm like, "Let me just go through all this stuff." But, at a certain point, I've run out of that. I'm always interested in learning new stuff myself, so machine learning has become a bit of that area that I've been teaching myself, and then making videos about. That has been driving a lot of the new topics.

And I do happen to get a lot of requests or suggestions from the viewers and from the community.

A good example of this would be the most recent thing that I did: Tic-Tac-Toe (https://www.youtube.com/watch?v=GTWrWM1UsnA). [It was a coding challenge], just making a simple interactive version of playing the game tic-tac-toe. I felt like, "I understand the game tic-tac-toe in my head. Maybe I've programmed it as an example in a class. This is something I should go in and do without thinking about in advance to show that process."

"Coding Challenge #149: Tic-Tac-Toe"

So, it's on a case-by-case basis. That's what I'm doing these days: picking topics that people suggest, which are either a standard topic that you might find in a computer science class, or otherwise a fun game or creative idea that somebody threw out there. That's how I approach these coding challenge videos, and those, ideally, range between 10 minutes and an hour, the sweet spot being 20 to 30 minutes. I'll do those during the livestream, but then I work with a video editor. We don't remove any of the actual writing of the lines of code. It's important, I think, for people watching it after the fact to be able to see the whole process from start to finish. But there's often a moment where I'll be coughing, or I went down a rabbit hole of trying to figure something out, which is really not important in the end. So that will get edited down. That's, these days, how I'm trying to think about these coding challenge videos.

SK Hi everyone. Welcome to createCanvas, a podcast about the Processing education community. I'm your host, Saber Khan, Education Community Director of Processing Foundation. [Intro is the same as above.]

This is the second part of our conversation with DS of Coding Train Processing Foundation and NYU ITP. If you missed the first part, I recommend you listen to that as well. You can find that and other episodes on the Creative Canvas SoundCloud and wherever you download podcasts.

By the way, these interviews are part of a new feature on our website, processingfoundation.org, called the Education Portal. The portal will have education materials and resources available for free.

You moved from Vimeo to YouTube, and YouTube is such a sort of cultural moment both of learning about the world, but also anxiety about what's happening here.

Does that enter into the Coding Train world?

DS It does and it doesn't. It enters into my thoughts a lot because I have two kids who are eight and 11. They spend time watching YouTube. I'm always trying to figure out how to manage that; and "should they be watching YouTube?" and "how many clicks away are they from inappropriate material or conspiracy theories?" I worry about that. So that's a concern of mine and, in a lot of ways, I might prefer to be on a platform, or have my own platform, that is less part of the giant corporate mess that is the internet.

However, for what I'm doing, I don't know how I would reach that same scale of audience, and it has offered me opportunities to generate revenue, which make doing this more sustainable for me. So on the one hand, I talk about working for Processing, doing that without any additional income, [and how that] works because I consider it part of my research practice.

The same could be said about making video tutorials. I could put them out with no ads on them; I do put them out with ads on them, and sometimes I take

the ads off, and sometimes I put them back on. I go back and forth between this depending on if I'm requiring them for a class. I'll demonetize them for like a couple of weeks, then I'll put it back on.

It's allowed me a couple things. One, is I now [at the time of the interview] have two people [as of this publication, it's now four] that I work with who I pay. Having the income makes that possible. It's also really a motivating factor for me. It helps me get excited about doing it in a certain way, because I know there's that benefit to me coming back from it as well. I'm conflicted about this because I'm in a very lucky, privileged position to have a full-time salary job at NYU, but I also think of it as, if I can learn how to do this in a sustainable way, that generates revenue, then that's something that hopefully I can model for other people who can do the same type of thing. The YouTube platform really affords me this possibility.

I have a backup of all of my videos. YouTube could go down, or YouTube could do something really terrible, and I feel like the irony here is, "Where is my backup of all my video files?" Oh, it's in my Google Drive. [laughs] That's also Google. So I don't know how to manage this.

SK [laughing] Yeah.

DS I'm kind of like handcuffed in this situation: it's of great benefit to me, and I really enjoy YouTube as a platform and the livestreaming, everything that comes with it, but I'm concerned.

Anybody who has thoughts about this, I would love feedback, or to hear from [you], to make sure that I'm doing everything in the best way that I can!

SK It also seems to have a very interesting, vibrant community of creators, especially in the area you work.

DS Right.

SK Have you met other peers doing similar work?

DS Yes. This has been one of the most fun aspects of doing this on YouTube. I feel like I have been part of a community—whether it's the ITP community, the Processing community, or open source—for years, but only in the last couple of years did I discover this other community of educators on YouTube. Most of whom are the most well-known ones [who] are not doing computer coding.

I've gone to some events; I went to something called EduCon, which is a gathering of people who make educational video content. I went to something called ThinkerCon (https://www.youtube.com/watch?v=UXq1VFWTQE0&feature=youtu.be), which was in Huntsville, Alabama, which was, again, a sort of conference and gathering of people. I've met a lot of really amazing, phenomenal, talented people who make science, math, and other kinds of educational videos on YouTube.



Learning from them, meeting them, and trying to figure out ways to do collaborations has been great. That is also an answer to one of your other questions that I forgot about, which is that a primary way I get ideas is [by] watching a lot of other educational YouTube creators and thinking, "Okay, so I just watched...." For example, the image of the black hole was recently published, [and] a very well-known YouTube channel called Veritasium, which makes a lot of physics and science videos, made a whole video about that image [which you can see here (https://www.youtube.com/watch?v=zUyH3XhpLTo&feature=youtu.be)]. Watching that video [I thought], "Is there a way I could code an example that would go along with it?" That's how I see my role.

There's a YouTube channel called 3Blue1Brown (https://www.youtube.com/channel/UCYO_jab_esuFRV4b17AJtAw), which does a lot of math and physics educational videos with animations and all sorts of amazing, beautiful explanations. My skill is never going to be making such beautiful, eloquent, concise explanations of really complex topics, but I can sit and struggle through coding one of those topics. In a way, what I can do, is be a companion service to some of these other YouTube channels, where people who are watching and learning and getting excited about a topic, could find a coding video that goes along with it.

SK I think that's really to another point about what coding offers. Someone's worked out the math and the science behind it to play with it. It's great to be in a coding base for a lot of people so that you can watch it happen or play with the variables and things like that.

DS Yeah and for me it's always been a way of understanding something that I thought I wouldn't be able to understand.

SK Maybe this is a good moment to talk about the thing you hinted at: ml5js (https://ml5js.org/). What is that project? What's happening there? How's it connected back to some of the other things you mentioned?

DS Yeah. So, ml5 (https://ml5js.org/) is a JavaScript library. It started for a couple of different reasons. One, is it started out of a grant that I applied for from Google (speaking of Google, everything in my life seems to tie back to Google!). Google has something called Google Faculty Research Awards, I think that is the name of it, that you can apply for if you teach at a university.

Dan's series of video tutorials on ml5.js comprises 11 videos (as of now) and teaches the concepts behind machine learning using the ml5.js library.

At the time, Google was coming out with something called DeepLearningJS, which is now TensorFlow.js, which is an open source JavaScript library for machine learning. I wrote a grant proposal to say we wanted to make beginner-friendly examples for people learning to code with things like p5.js, to be able to use TensorFlow.js and be able to learn about machine learning and creative coding.

That ultimately became more than just a set of examples. It also became a library called ml5, which is just a wrapper around TensorFlow.js that provides access to a lot of the functionality, but without the requirement to do some of the lower-level things you need to do when you're working with TensorFlow.js directly. That library exists as a standalone project developed at ITP. Most of the contributors are ITP students or researchers, but there's been a lot of outside contributors as well.

This recent video introduces Teachable Machine 2.0 (from Google Creative Lab). "Train a computer to recognize your own images, sounds, and poses," using p5.js with the ml5.js library.

Two students from Parsons this year joined the project and then helped design the new ml5.js website. That project, it's kind of grown and sits alongside, almost as like a sibling project to p5. It's not officially part of the Processing Foundation, but there are a lot of the same people working on both of those libraries and p5 and ml5 work well together.

I'm teaching a new class this fall, and I'm hoping to use ml5 and TensorFlow.js and something else also called RunwayML, which is another separate project, but those three platforms as the primary engines for this course, in which case I'm hoping to make a lot of new video materials and make a lot of new tutorials about [them] as well.

SK That's great. It seems like there's a way to understand what it is that you do, which is this desire to create stuff for your classroom that has led you down many different roads, the Processing Foundation, Coding Train, ml5.

DS Right.

SK What's your hope for the direction Processing Foundation's headed in? Are there things that you're looking forward to at Coding Train? I'd love to know what the future holds.

DS That's a great question. One thing that, it's not really to answer your question, but one thing that just struck me, that's come into focus through this conversation, is I feel like the common thread throughout all of these things, in a way, is that last mile (or kilometer, I try to use the metric system whenever I




FIG 002–04 Map showing the 100+ registered node cities of Processing Community Day Worldwide 2019.
Google map with markers all over the world.

can!). There are so many things out there, like TensorFlow.js—to name the one that's most recently talked about—that do all of these amazing technical gymnastics, and they shouldn't necessarily. You can't do it all. If you're the people developing TensorFlow.js, you have to focus on what you are focusing on to make that thing work, but there's this little gap of somebody who is a high school student, or a lifelong learner in their sixties who is sitting on the internet and knows a little bit about coding, and cannot figure out how to get what they're working on to attach to this new thing. I feel like, for me, my videos try to do—it's not technically complicated work necessarily—it's more community work and documentation work, and being thoughtful about the language you're using to bridge that little gap there. I think the p5 Web Editor that Cassie Tarakajian leads [here's a short video intro (https://www.youtube.com/watch?v=dtHxDggkBYc&feature=emb_logo)], is a project that does this as well, because I've encountered so many people who say, "Okay, I sort of get what you're talking about in these four lines of code, but I don't know how to put those lines of code on my computer and run them."

SK Mm-hmm [affirmative].

DS Being able to do that work is so fundamental and crucial. Thinking back to your question, my hope is for more of that, and especially more that's targeted towards communities, of people who have historically been left out, or who don't have access; or materials haven't been written with them in mind; whether they're not seeing themselves represented in the people who are presenting this stuff, or there's just been no one who's come to their city or community to do a workshop.

That's one of the things that I know began with Processing Community Day (https://processingfoundation.org/advocacy/processing-community-day-2017-2019), specifically Processing Community Day Worldwide (https://processingfoundation.org/advocacy/processing-community-day-2020). Having these events in cities that are places that aren't just New York and Los Angeles, or other similar international cities, but really being able to reach beyond the people that I'm used to seeing and talking to.

I think we have to find ways of bringing more people to the table, not just like, "oh, we're running workshops in these cities," but that there are people who are planning which cities, who are from those cities, and from those communities. That's really what I'm hoping for. I think YouTube does that in some ways, because it reaches people in international audiences. If you have a phone, but don't have a computer, if you have an internet connection, or a mobile connection, you can find these videos and learn about this stuff. But it doesn't reach people in other ways that having a local meet-up can. Or something that's funding people to work on open-source contribution, paying them to do, like, a two-week project, and mentoring them. That's not what Coding Train does.

I feel like the university, NYU as an institution, offers something to a certain audience, YouTube offers something different to a different audience, and Processing Foundation also, hopefully, can be a place that can bridge a lot of those gaps of audiences who are being left out of those two things as well.

SK That's great. One of the reasons we're interviewing you aside to hear all these things, is to launch our Education Portal (https://processingfoundation.org/education), which is on the Processing Foundation website, to help people get started by themselves or in their classroom.

From all that stuff you mentioned, is there a particular place [someone] should start? Is there a particular video or a playlist that you recommend?

DS There are many places on the internet that you can find to get started. I'm always speaking about my stuff because it's on the front of my mind. I don't want to recommend it over the other wonderful tutorials and starting places that are out there. I do have a playlist. It's called Code! Introduction to Programming with p5.js (https://www.youtube.com/watch?v=HerCR8bw_GE&feature=youtu.be).

This video series teaches the basics of programming with p5.js for beginners, and includes seven tutorials so far.

That's the playlist that is meant to be for someone who's never done any of this before. The only requirement to follow along is that you have a web browser, because it's assuming you're going to use p5 Web Editor. At present [the web editor] doesn't work on mobile or tablet very well (maybe in the future it will). So that's the starting point.

One of the things that I'm most excited about with the Education Portal, is a way of aggregating more of these, because I know there are other people who either have written tutorials or video tutorials that are for total beginners. Being able to have a repository of some of that stuff for people to find things more easily is super important.



Also, I think what's really important that we've been doing a better job with, especially with the work that you're doing, is talking to teachers. Because when I found Processing in 2003, it was primarily being used in graduate school environments, art school environments. Especially with what Lauren McCarthy and the p5.js community have built being JavaScripted in the browser, again, and even Processing wasn't that originally, it wasn't JavaScript that was in the browser originally, it's finding its way into K through 12 education. That wasn't the expertise or the experience of a lot of the core contributors and developers of processing in p5.js.

There are things that a teacher will say to me, like, "If this was in the web editor, this would solve all these problems we have." It seems obvious after a person has said that to me, but I never thought of it beforehand. Being able to have those kinds of conversations and feedback from teachers, and having them be part of the development and design process, through the Education Portal, is a really important thing.

SK Great. Thank you so much. Thank you for joining createCanvas. Once again, I'm your host Saber Khan. createCanvas is produced by Processing Foundation and supported by the Knight Foundation. Our editor is Devin Curry. Special thanks to the Processing Foundation board and staff.

PER 003

# Making p5.js, 2021,

Lauren Lee McCarthy.

They want us to think technology is a black box. Accept the terms of service. You can't know what's inside. You can't modify it. You certainly can't make your own.

We didn't believe it, so we made p5.js. Making p5 meant starting with deep uncertainty and many questions. Becoming comfortable with not knowing, with making mistakes, with always being in a process of learning. Following each question until it led us to many more.

*How do you get started with something new when you're scared?*
*How do we ask for help?*
*How do we think about making a tool?*
*Which users do we prioritize?*
*What assumptions are we making about them?*
*How do we organize our code?*
*Can we value every type and size of contribution?*
*When do we trade performance of our tool for legibility of the source code?*
*How do we elevate the role of documentation?*
*How can we teach people to use the tool?*
*Which language are they speaking?*
*Which words are we choosing to use?*
*How do we make decisions with a large group of people?*
*Can we make space for many different people to lead?*
*How do we acknowledge the endless amount of emotional labor required?*
*How do we support each other?*

We were artists, designers, coders, writers, organizers, teachers, students, beginners. We were contributors. We could see that our open-source project had so many more needs than just code. We needed people to write tutorials and documentation and website text. To answer questions on the forum and respond to issues on GitHub and answer emails. To create curricula and examples and livestreams and teach people how to use it. To reinterpret the project in other languages and cultural contexts. To test and build out web accessibility for disabled users. To work with their local communities to organize workshops, meetups, exhibitions, community spaces, community days. To share code and ideas through their artwork. To organize contributor conferences and draft the p5.js community statement. To work through conflicts and different opinions. We needed people who could recognize the time this work requires, and the need to pause sometimes.

No tool is neutral; each is built with the beliefs and biases of its creators. We were making a tool for creative expression, so it felt especially urgent to have a wide diversity of perspectives contributing to its making. We were constantly thinking about community, and the barriers to entry. How do we welcome someone? How do we keep updating our understanding of access?

Who is being cut out by the "getting started" tutorial that's too confusing, by the tone of communication online, by the use of English language, by the lack of a computer or internet connection, by not seeing people that look like them hold roles in this community, by content that is inaccessible to disabled people, by not having the financial means to volunteer time?

We felt the uncomfortable contradictions of trying to make this project within a capitalist framework as we dealt with the realities of a lack of time and resources. We got things wrong and got hurt and got burnt out. We reminded ourselves: *It's always an option to throw our laptops into the ocean and bow out, but we have chosen not to*. Making with p5.js was believing you could make things with other people. We could keep trying, iterating. We were building on decades of trying and learning from teachers in other communities. Processing, School for Poetic Computation, openFrameworks, Design By Numbers, Arduino, Wiring, three.js, POWRPLNT, Afrotectopia, PyLadies, Design Justice Network, and many others.

But we also wanted this to be a project where you *could* bow out. As much as we prioritized making it easy to enter the project, we wanted it to be just as easy to exit. To open space for others, and open space for yourself to do new things. To be a part of the community even after you've signed off.

We began rotating the project lead. We held an open call process where we asked how this position could support learning for the lead as well as the community. Could the community provide collective mentorship? Moira took over, and then passed it on to evelyn and Qianqian. They have each brought new questions to p5.js, stewarding it on a path forward from the 1.0 release that explicitly centers access in every new feature.

Through the years working on p5.js, I realized I am much more comfortable writing *we* instead of *I*. Because every part of this project feels like something I learned from someone else. A million little gifts from different people merged into a library and a community. Yet there were also moments where I felt alone and unsure, and I know many others felt this, too. p5 is remembering you have other people you can ask to work through the questions with you. We set our own terms. p5 is everything we learn from each other when we make ourselves open.



PER 004 title? #58, 2020, Casey Reas.

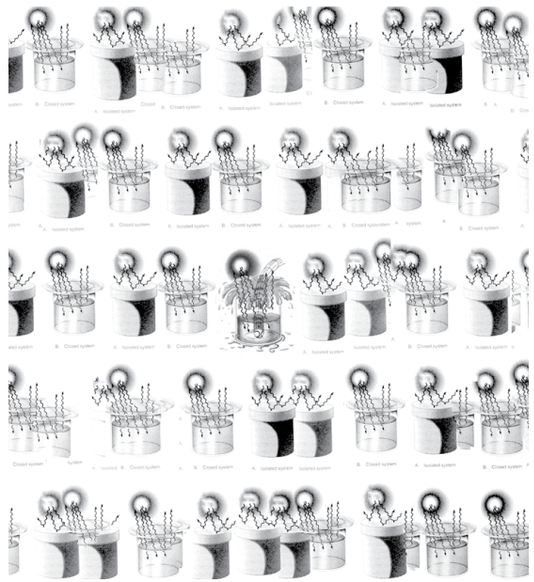


FIG 004–01 Content Aware Collage, Alice Mira Chung. Algorithmically generated black and white image of beaker of water with sunlight, showing the open and close isolated systems in thermodynamics through 2D graphics.

I was around for the birth of p5.js and I'm around now for the 1.0 release over six years later. My greatest contribution to the project was to keep my nose out of it. Through Lauren's guidance and energy, p5.js has grown into something completely different from its parent, Processing. Both projects share the same domain and emphasize community and learning, but p5.js imagines a new, refreshing approach to situating coding within the arts and vice versa. Processing's core is rooted in the culture at the turn of the century and p5.js points beyond 2020.

As I was thinking about the p5.js 1.0 release, something brought be back to the early days—to 2013. Some email searching to jog my memory led to a GitHub issue started on August 29 with the name "title?" The text that follows below is an edited discussion about how to name the nascent project that would become what we know as the amazing p5.js.

The cast of characters are Evelyn Eastmond (@evhan55), me (@REAS), Lauren McCarthy (@lmccart), Chandler McWilliams (@brysonian), Erik Blankenship (@jedierikb), and Dan Shiffman (@shiffman). The full discussion is here (https://github.com/processing/p5.js/issues/58).

title? #58

evhan55 opened this issue on Aug 29, 2013 · 19 comments

evhan55 commented on Aug 29, 2013

@lmccart

What is the official name of this library?

ProcessingJS

Processing-JS

Processing-js

processing.js

processingjs etc....
How are we differentiating from Processing.js?
Keeping track of Processing, processing.org, Processing.js and processingjs.org and now this new library is going to be really confusing to many people.
Perhaps something to bring up in the September 4th/5th meeting?

REAS commented on Sep 11, 2013

Right now, I'm leaning toward the minimal "P5" as the project name. I'm still fond of "Ciel." I'm curious about what the RISD grads might come up with.

lmccart commented on Sep 12, 2013

hm, it is an abbreviation for perl 5, but I think not too much googling overlap there.

brysonian commented on Oct 10, 2013

p5js has a lot of results, and p5js.com so that might not solve the confusion. I can't come up with much either though. I did like the name Cing for that cpp project that was going on. Maybe something similarly odd/nice sounding? jxing?

jedierikb commented on Oct 10, 2013

p5j5 ?
has a nice r2d2 ring to it.

REAS commented on Oct 10, 2013

My daughter says "3PPO" instead of C3PO. I wonder what we can learn from that?

brysonian commented on Oct 10, 2013

ppppp.js

shiffman commented on Oct 10, 2013

i think if you put a P in front of it, it will be perfect.

lmccart commented on Oct 11, 2013

👍

evhan55 commented on Oct 25, 2013

The RISD grads haven't really chimed in, unfortunately. They are still getting a grasp of the difference between Processing, Processing.JS and this new version, and none of the students in my section have suggested alternatives.

I am starting to lean towards "P5" but not exactly fond of "ppppp," it seems like a mouthful and beginners might wonder why. Also still really like 'Ciel' or anything completely different and fresh, but since this library builds on the Processing API so closely, the tie to 'proce55ing' seems nice.

Although a quick Google search... http://www.sojamo.de/libraries/controlP5/

Is P5 used in other places to refer to Processing?



brysonian commented on Oct 25, 2013

It was for years, not as much anymore. Something simple like draw.js might be nice too.

REAS commented on Oct 25, 2013

... After receiving so much negative feedback about the name "Processing" over the years, I think common usage language names are tricky. However, maybe the advantages of the simple "draw.js" outweigh these negatives. @evhan55, people naturally named Processing libraries with "pro" and "p5," but we discouraged them because it made them feel "official" rather than "contributed." I don't think this is an issue for your project.

lmccart closed this on Nov 1, 2013

jedierikb commented on Nov 1, 2013

And... The title is????

lmccart commented on Nov 1, 2013

I believe we have settled on p5.js. @evhan55, is this right or is this still tentative?

evhan55 commented on Nov 1, 2013

p5.js :)

PER 005 why i love p5.js and how did i get here, 2020,
aarón montoya-moraga.

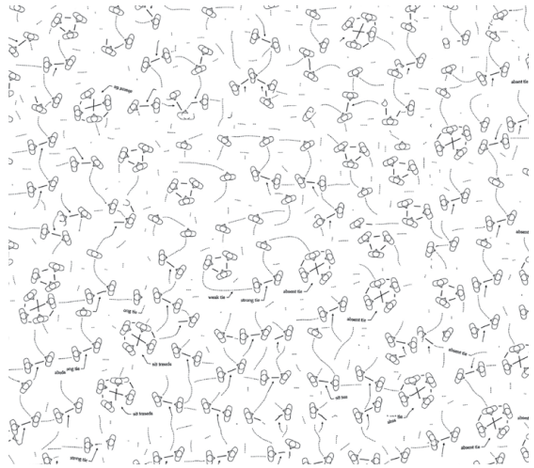


FIG 005-01 Content Aware Collage, Alice Mira Chung. Social network model illustration depicting interpersonal ties through two-dimensional lines, circles, dots, arrows, cylinder and words, e.g. absent tie.

p5.js is my favorite community, people, software, tool
let me first explain my journey so far, and highlight great people i have met

2011 i used to be a struggling electrical engineering undergrad
professor Rodrigo Cádiz who invited me to a computer music workshop
i learned we could use code for arts, and met a beautiful community of media artists

2012 i started learning Processing and still didn't understand what free libre open source meant

2013 artist Margarita Gómez showed me openFrameworks and i couldn't figure out how to use it

2014 performance artist María José Contreras hired me for my first job as a programmer for arts
i met Delight Lab and they taught me some video mapping, i was hooked

2015 i was accepted to NYU ITP, a two-year graduate school media arts program
it was the first time they taught p5.js to first years instead of the usual Processing
we learned how to program cool wacky interactive websites and how the internet worked
since then i haven't stopped using p5.js :)

2016 i took my favorite class at grad school, Lauren McCarthy's Performing User
check it out, it is amazing https://itp.nyu.edu/classes/performinguser/syllabus/
at engineering school i had been taught programming as an useful skill for making money
at art school i was inspired by the creative potential of computers
i became convinced artists and society would benefit from using code as a creative medium
one of my biggest privileges has been learning English and through that, access to education

but if you don't know English, i guess it's hard to know what you're missing, like p5.js
i wished kids from Spanish-speaking communities would have no barriers to their arts education
i got a grant from the Processing Foundation that summer
my proposal was to translate to Spanish the p5.js website and book, and teach workshops
the website i18n had been started by Maya Man and i learned so much from her work
i did rough translations of the book and website, but i felt i should do more
2017 after grad school, i spent that summer finishing the translations
Lee Tusman mentored me, Daniel Shiffman and Rune Madsen helped with code for the book
Taeyoon Choi translated the artwork, Tyler Yin and Casey Reas made the book a reality
i explained to Dorothy Santos, Lauren and Casey that the book should be online,in both pay-what-you-want PDF and optional physical copies
and that all the source code should be available online for everyone to contribute
they were encouraging and made this dream a reality
here is the source code https://github.com/processing/p5.js-getting-started-es
and the PDF and link for ordering physical copies https://processingfoundation.press/
i was hired by NYU ITP to stay for one academic year as a research resident
with four of my best friends we started a media arts school called CODED Escuela
Camila Colussi, Natalia Cabrera, Guille Montecinos, Christian Oyarzún
together we taught p5.js in Chile and fostered the Chilean media arts community
2018 i represented Processing Foundation in Argentina for the +CODE Festival
people were so excited to learn p5.js and celebrating media arts
2019 i helped organize and taught at the Processing Community Day in Quito Ecuador with friends Gabriel Andrade, Ava Huang, Sharon Lee De La Cruz, Andrés Colubri
i joined MIT where i am creating open-source tools for arts and learning digital rights
i attended the the second p5.js contributor's workshop and worked on the global track with Yasheng She, Qianqian Ye, Kenneth Lim, Luis M-N, and so many others!



we helped each other figure out the role of p5.js in a worldwide context and discussed the sociopolitical aspects of software, of media arts, of sharing
2020 in two months i will go again to PCD in Ecuador with Francesca Rodríguez-Sawaya
shout out to Saber Khan for organizing Processing Community Day 2020
everyone i have met through p5.js and Processing has been inspiring and caring
they advise me to please don't burn out, to please self care, and enjoy the process
one of my favorite moments ever was when Francesca tagged me on a picture of the p5.js book in Spanish being used at schools in NYC to learn media arts
thank you p5.js friends for caring about each other
thank you for making this world a happier more ethical place with more art
:)

# PER 006 Oddkins, 2020, Taeyoon Choi.

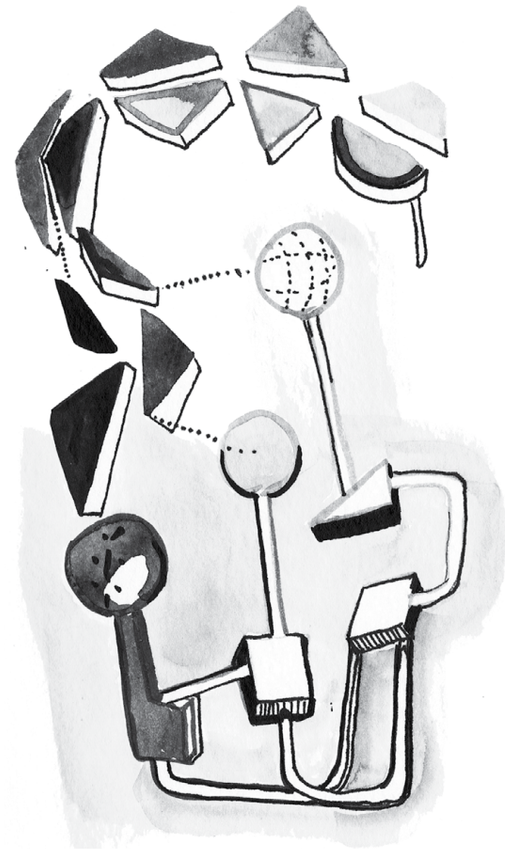


FIG 006-01 This drawing by Taeyoon Choi was originally commissioned for the Getting Started with p5.js book by Lauren McCarthy, Casey Reas, and Ben Fry.

Computer generated illustration depicting networks done by Taeyoon's hand drawings, showing the interconnectedness of computers and humans.

I'm an oddkin of p5.js contributors. Oddkin is a word Donna Haraway uses to describe an unlikely intimacy. The word oddkin captures the essence of friendship I have in personal life, intellectual kinship with my students and peers at the School for Poetic Computation and in the geographically diverse nature of the Processing community. I'm bilingual; it's natural for me to try to translate the word into Korean. I looked up kin, which is 친척 (親戚 in Chinese). I don't think the word oddkin is translated into Korean. I asked if anyone would help translate it, and the media artist Masayuki Akamatsu suggested 기묘한동류 (妙な同類 in Japanese). It doesn't have the earthy, crittery feel of oddkin, but it's close enough. In this essay, I take an opportunity to think about you, my oddkins.

Dear oddkins. I've been lucky to participate in and help with the p5.js contributors conference at the STUDIO for Creative Inquiry in 2015 and remain a fan of the project. Some of my colleagues and students say they are trying to "democratize" technology by making tools or resources. I'd like to challenge the implications created by the use of the word "democracy" in this context. "Democracy," unlike "accessibility," implies a fair distribution of power, representation, and equity for everyone, but is complicated by ongoing questions of citizenship. How many times have you heard a hacker say, "I made this open source tool. Now everyone can do what I do!" What they don't say is that to use it, you need to be technically fluent, accepted as a technical person, and have access to computers and networks. In reality, tools and resources alone cannot be democratic.

What the p5.js project has done differently from the beginning is demystify stereotypes of who is seen as technical and/or artistic—the myth of the lone engineer capable of creating a powerful framework—and create a technical community where non-technical contributions are valued equally. This approach is truer to the reality of software and hardware development. There are almost always non-technical contributors who are integral to the success of any technology, including but not limited to writers, educators, testers, designers, and Quality Control professionals.

The questions of whiteness in the technology field remains. I've attended numerous software and hardware conferences over the years. Some were highly exclusive and self-selective, others were open and inviting to the newcomers. In the U.S. and Europe, the conference attendees were dominatingly white males. I could write a book about the troubling experiences I've had and witnessed at tech conferences. Older, established, white male engineers and academics making rude comments to female, gender non-conforming, people of color might need a few chapters. Young tech dudes promising to solve the world's greatest challenges with code. "Introverts" who are actually just snobbish, escaping to their devices to avoid human contact but invasively stalking women with confidence. The most insidious forms of racism that happen offstage. How our speaker badge isn't enough for us to enter the speakers' lounge, how white bartenders treat us with offhand racist jokes, how we are tokenized for photo opportunities and a shout out on diversity statement. They can all get a chapter of their own in my book.

To survive in white spaces as a person of color, I learned to adapt to the expectations of others at first. Once I felt grounded, I leveraged my place to give voice to the others. Speaking up my truth made some people uncomfortable and I may have lost some opportunities. Speaking up brought me the respect of others, especially those who hold the power in white spaces.

How does one speak up to challenge the establishment? I think nuance and candor are essential to start a conversation. No one likes to be criticized, but everyone loves to be understood. When I was learning to code, a friend told me "treat your computer like your younger sibling." This approach of care and play is essential in exploratory and poetic computing. I think a similar approach is necessary to navigate the tech community. As the p5.js project grows in its scale and diversity, I want to ask, "who do we make our oddkins?"



PER 007 p5.js as a Family, 2020, Aren Davey.

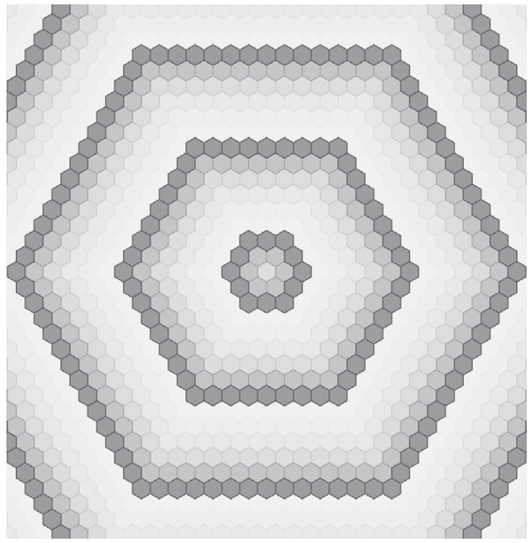


FIG 007–01 Image from the p5Grid library by Aren Davey: https://github.com/aahdee/p5grid. Algorithmically generated illustration of a centered radial hexagon sequence covering the entire canvas with a white to light gray color rhythm.

*Creating* is a verb that I find describes my childhood and teenage years. By designing, crafting, and building, I morphed my wild imaginations into tangible objects. Intricate patterns and inventions sprung from my daydreams in my high school classes. I honestly wanted to choose graphic design or architecture for my career path, but since I did very well in the maths and sciences, my family pushed me towards those fields instead with the reason that they didn't want my talents to go to waste. Since I didn't want to disappoint them and I actually had an interest in engineering and scientific research, I promised that I'd go to college for a degree in that field and almost completed every math class my high school offered. I also arrived at school almost an hour earlier for extra art classes, completed many studios, and did independent art study. Whether my intense motivation was to expand my interests or an act of small defiance is up for discussion, but my art practice has always been seen as an afterthought. Thus, I started to innately see it as an afterthought as well.

Quite frankly, the p5.js community is not a group that I expected myself to be in. Having been intrigued by the concept of Processing and p5.js, I used the platform for my school work and expected that to be the extent of my involvement. Then through a string of chance happenings, I ended up at the 2019 p5.js Contributors Conference. I immediately felt out of place because I didn't really feel like I deserved the title "contributor". Like, I was 21, still in college but hanging on for dear life, and felt invisible compared to the other contributors who had a pretty substantial Twitter presence and were speaking at or hosting large conferences like Eyeo, maybe SIGGRAPH. I'd be entirely grateful if I could just see the door to the SIGGRAPH exhibition hall. One of the other attendees happened to be a visiting professor who was teaching a class that I was enrolled for the upcoming semester. All of these people were doing great things that were impacting the p5.js community and beyond so I didn't feel like I deserved to be on

the same level as them. Creating was only applied to projects that I finished for a grade, so I hadn't ever done anything that I designed and executed myself. Thus, I felt pretty lost and doubtful if I could ever create anything for this community.

But after receiving the feedback that I got from pitching an idea that I had sitting in my head for a few months, creating a very dirty alpha for that idea, and seeing a visible, tangible, actual product sprout from that, I finally felt like a contributor. I saw that my art practice was taken seriously by others and there was genuine interest and need for it. Both of my interests, software development and fine arts, were seen of equal value and status. One wasn't inherently better than the other, but instead they are both appreciated and valued in their own way. The p5.js community is a place where that holds true, so I'm glad that they have accepted me and that I have the chance to give back to them.

I sometimes prefer to call the p5.js community a family instead. It's more tight-knit than other online developmental communities and its members are genuinely interested in the progress and evolution of others. I might not have the same amount of experience as most of the members, but that doesn't make me less of a contributor. Everyone here is ready to assist others who need it and ensure that they go to places that they have never thought of before. This family exposed me to things I wouldn't have seen without their help and has helped me develop as an artist and a contributor. No longer is my art practice felt as an afterthought; it is a significant piece of myself that is worth exploring, developing, and investing in, and I'm very glad the p5.js family helped me come to that conclusion.





# p5 Fellow Experience, 2020,

Stalgia Grigg.

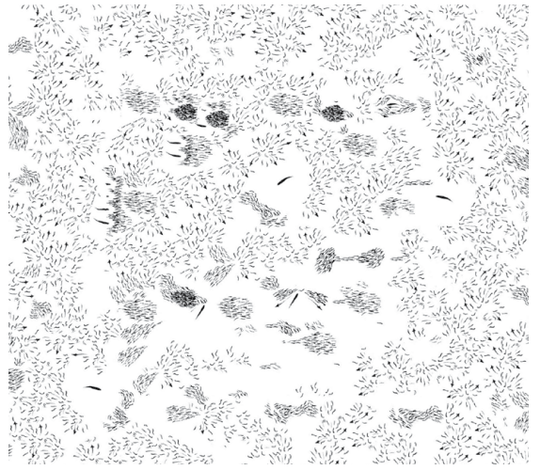


FIG 008–01 Content Aware Collage, Alice Mira Chung. Aggregation of short arrows, curvy lines, and strokes with an air of fluidity and garden-like illustration. Patterns exhibited are based on emergent behavior patterns of fish schools.

During 2019, I served as a fellow for p5.js. During the year I was given the opportunity to operate as a co-leader for the pro ject. This meant that I was given agency and responsibility in everything from maintaining the open-source codebase and developing a vision for future governance to designing the documentation for welcoming new contributors and honing the release roadmap based on the community mission. I have been a contributor to p5 for several years now, but it wasn't until my experience with this fellowship that I finally understood the significance of the example that p5 is setting.

My path as a software developer has been closely aligned with p5 from the beginning. My first experiences coding were with p5. I first encountered a for-loop in the late nights poring over the online reference and examples for the p5 library as I made small experimental creative sketches. This began a lifelong dedication to and love for the craft of programming. That dedication led me to become a contributor to the project. I learned how to work on an open-source project through my engagement with p5. I can say confidently that if it weren't for the generous patience and understanding of the p5 community, I would've been too intimidated or discouraged by my early mistakes to push through and achieve confidence in my status as a contributor. The p5 community offered a sense of comfort in learning through doing. This meant that I was able to learn a more advanced approach to software development and library architecture design, but it also meant that I learned by example how to be a steward for a web community. This process of learning eventually culminated in my experience as a fellow.

As a fellow, the responsibility that I was given gave me insight into the intention that forms the overarching shape of the project. p5 offers every possible avenue for involvement and empowerment. It offers a vast library of supporting educational resources so that people can learn how to code by making things. It offers a site for consideration and discussion of how a radically inclusive software project can be a political act. It offers an intimidation-free, encouraging space to experiment with open-source

contribution. It offers a flexible vision and malleable governance that empowers any contributor to reimagine the future of the project. All of these different angles for engagement are made available to anyone who shares a passion for p5's mission. This forms an entire ecosystem that works tirelessly to empower people who have been historically left out of the conversation of how software tools and the open web are built. My deep engagement with the p5 community led me through all of these different angles over time, giving me the gift of a path from self-taught coder to community-leader, software advocate, and mentor.

My experience as a p5 fellow has been formative in my understanding of how I want to interact with the world as a software engineer, tool-maker, developer, and activist. The approach that p5 exemplifies, community-forward and porous, is one that will fundamentally inform my future work organizing projects that fight for a radically inclusive and definitively open web. I'm glad that they have accepted me and that I have the chance to give back to them.

I sometimes prefer to call the p5.js community a family instead. It's more tight-knit than other online developmental communities and its members are genuinely interested in the progress and evolution of others. I might not have the same amount of experience as most of the members, but that doesn't make me less of a contributor. Everyone here is ready to assist others who need it and ensure that they go to places that they have never thought of before. This family exposed me to things I wouldn't have seen without their help and has helped me develop as an artist and a contributor. No longer is my art practice felt as an afterthought; it is a significant piece of myself that is worth exploring, developing, and investing in and I'm very glad the p5.js family helped me come to that conclusion.



PER 009 How to Write Non-Violent Creative Code, A WIP Introduction, 2019, Olivia McKayla Ross.

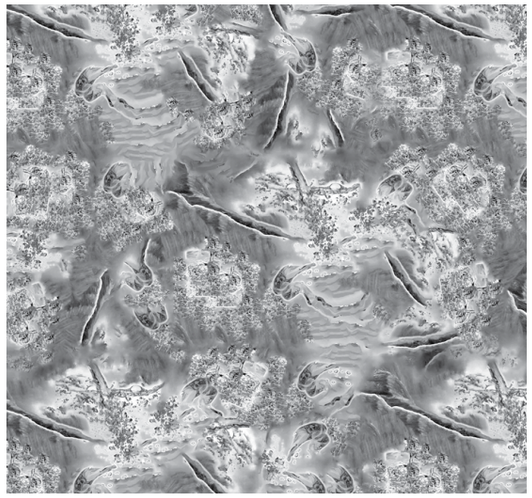


FIG 009–01 Content Aware Collage, Alice Mira Chung. Gray background with monochromatic brush-like strokes created from an InPainting algorithm using a blend of stock photo images of water.

During the 2019 p5.js Contributors' Conference, a group formed to discuss how radical inclusion, decolonization, and a decentering of dominant communities could be further integrated and emphasized in p5.js as a toolkit and a community. We acknowledged and discussed the ways in which p5.js is built upon (and benefits from) colonial projects like the English language, the internet, and their related infrastructures.

*It's always an option to throw our laptops into the ocean and bow out, but we have chosen not to.*

We agreed to strategize how we as practitioners and toolmakers can reindigenize creative coding, starting in the areas where we have the most agency—ourselves and our works.

*"Reindigenization is not a repeat of the past, it is the revolutionary/evolutionary RETURN to ways of life with and for the land."*

Drawing from a collective list of references, we've drafted a set of guidelines, questions, and prompts to help us and others reflect and refine our work in this spirit.

The term "nonviolent creative code" was proposed to guide this work. In this context, we use the term violence to include actions that keep us from the earth, from ourselves, and from each other.

Honor the Land

Who is the community that lives here? Who is the audience of your work? Are those two populations different or the same? Why?

Are you living or working on occupied land? Who is the indigenous population where you are?

Is this art taking up material space? How have you considered managing the waste generated by this project?

Consider the materiality of the server, the data center, the cable, the rare-earth mineral mine, the factory, the oil field, the air conditioner, and the other things that are holding up this practice. Where can you decouple those links?

What is lost when reduced to data? What is there about your place that is not saved in the online database or archive? How does it feel here?

Honor the Body

There is an assumption that writing and coding are leisurely because you're sitting, but there's also posture issues, carpal tunnel, etc. Take breaks, drink water, protect your eyes, etc. Honor your body cues as data.

Whose bodies have contributed to your work? Who answered that question on Stack Overflow, who made this font, who assembled your computer? How are those bodies visible or invisible?

Does your work respect all participants' body sovereignty? Some participants may not wish to be photographed, recorded, scanned, or touched. How are you honoring these sacred boundaries?

Honor the Small

Does your project need to scale? What would your project gain from specificity? De-emphasize growth as a goal of a project. Instead, think of growth as one of many design principles.

Can you produce or share work on a large platform (Twitter, Instagram, etc), without doing work for that platform? How can you decouple your online practice from tech giants?

Honor the People

How do you prioritize consent in online space? Does engaging with your work require the participant to sign a terms of service? Do you control that terms of service, or is it in the control of a company or organization?

How do you protect your community? If you are making work that uses or stores information about the data, interactions, or bodies of others, how do you keep that safe?

Pay attention to when you use language like "mining" data. How can you maintain an explorative, rather than exploitative, relationship with the data collection process?

Does the code you're writing/using support structures of violence? Do the toolkits you use align with the values of consent and dignity?

What is the offboarding process for someone who would like to stop interacting with your work? What can they take with them? What do you do with the things they have left behind (data, feedback, etc.)? Can people take breaks with the option of returning without penalty?

Does your project use design patterns that mirror cycles of addiction (e.g. copying the slot machine)? How can it promote healthy, balanced relationships with technology?

Honor the Exchange

How can your work prioritize community and relation?



Dissolve hierarchy in your teaching/learning by respecting wisdom in all its forms. New participants always have another expertise.

Is the code you're writing intentionally or unintentionally so hard to read or obscure that others cannot learn from or contribute to it? Why?

PER 010 p5 Access, Keep it Coming! 2020, Claire Kearney-Volpe.

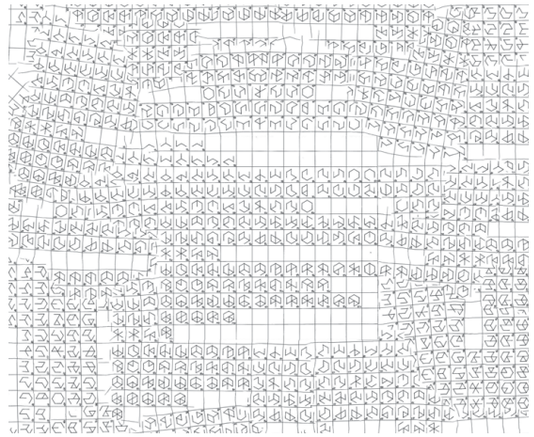


FIG 010–01 Content Aware Collage, Alice Mira Chung. Variations of a 2D cube with missing lines showing its openness all placed on a grid covering the entire white canvas. This illustration is computer generated from the diagram "Variations of Incomplete Open Cubes" by Sol LeWitt.

In 1996, the World Wide Web Consortium (W3C) formed the Web Accessibility Initiative. This initiative is responsible for creating what many people know as the Web Content Accessibility Guidelines (WCAG)—pronounced "wuh-cag," or "w-cag." WCAG is a series of design and development recommendations for making the web more accessible to people with disabilities. These guidelines are intended to support the creation of accessible *content* for persons with a variety of disabilities, including deafness, hearing loss, learning disabilities, cognitive limitations, limited movement, speech disabilities, blindness, and low vision (WCAG, 2019).

WCAG's cousin ATAG (Authoring Tool Accessibility Guidelines) are lesser known, but are intended to support accessible creation tools. Just as important as consuming content, authoring tool accessibility helps put means of production in the control of many, and democratize the creation of new media and technologies.

One super cool thing that digital accessibility principles give *everyone* is a framework to effectively communicate information across multiple modalities. It's also an opportunity to experiment with input and output types.

P5 Access Work

Over the last few years, p5.js has made great strides toward becoming a more accessibility-aware open-source community. A community that not only promotes and incorporates best practices, but pushes the bounds of communication and creativity.

The p5.js community took one of its first steps toward digital accessibility when Taeyoon Choi facilitated Signing Coders (http://taeyoonchoi.com/soft-care/signing-coders/), a "series of workshops on creative expression with computer programming, art, and poetry which focuses on reaching out to youth who are deaf or hard of hearing" (Choi, 2017). This initiative was p5's first and only educational initiative for accessibility designed from the beginning as an inclusive learning experience, with real-time transcription and ASL interpretation.

Shortly after this, Atul Varma (https://github.com/toolness) and I began teaching workshops on web development (https://medium.com/processing-foundation/p5-accessibility-115d84535fa8) who incorporated prototypes of tools to make learning and using p5 more accessible to people who are blind and use screen-reader technology. This work led to the creation of an accessible web development curriculum, and a better understanding of ways to support the accessibility of p5's visual outputs.

The Processing Foundation, true to its commitment to making coding a more accessible and inclusive practice, was generous to support the fellowship work of Mathura Govindarajan and Luis Morales-Navarro (https://medium.com/processing-foundation/working-on-the-p5-accessibility-project-58a781575400). With the help of a team of dedicated contributors, including Josh Mielle (https://twitter.com/berkeleyblink?lang=en), Lauren McCarthy, Cassie Tarajakan, Antonio Guimaraes, Elizabeth G. Betts, Mithru Vigneshwara, and Yossi Spira, Govindarajan and Morales-Navarro endeavored to implement a system for making the visual outputs of p5 more accessible to people who are blind. This work led to the development of a p5.js accessibility library and work to make the p5 web editor more accessible.

To Do

The road to a more accessible p5.js is a long one, and will take the efforts of our whole community—not just those who develop p5.js as a library, but those who teach it, document it, and develop project work. During the p5.js Contributor's Conference in the fall of 2019, the p5 community welcomed Sina Bahram to contribute and advise on the future of p5 Access work. This yielded the following working areas:

Educational work to promote awareness and skills

*We need to tighten up our documentation of p5's accessibility features and library.*

*We need to develop tutorials and trainings around web content accessibility for distribution amongst contributors and educators who use p5 in workshop/classroom settings.*

Tactical Work to enhance the accessibility of the p5.js library and documentation

*We need to make it clear in our documentation work what has been done, how to contribute (best practices doc), and what work is in the queue.*

Research to explore the dynamic generation of multimodal description of primitives and composites of primitives in a static scene to facilitate access for persons with disabilities.



PER 011 What Does it Mean to Be a Contributor?, 2020, Luis Morales-Navarro.

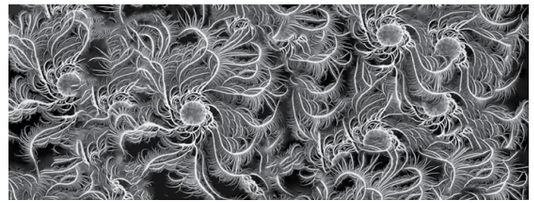


FIG 011-01 Content Aware Collage, Alice Mira Chung. Computer generated black background with white curves creating a botanical three-dimensional graphic quality from complex patterns by the Paencibacillus vortex bacteria colony.

Four days into the p5.js Contributors' Conference 2019 (https://p5js.org/community/contributors-conference-2019.html) I realized that I was not sure of what it meant to be a contributor. After the conference, amazed and intrigued by the diverse ways in which people participate in this project, I started asking questions, having conversations about the contributor identity with members of the p5.js community, and thinking through how the contributor identity fits into wider conversations about participation within FLOSS. As Johanna reflects (https://medium.com/processing-foundation/belonging-in-the-mess-3d3ad0577499) in p5 and Processing, open-source is less about writing software and more about how software is written. The p5.js contributor identity is constructed in response to the status quo of software development communities. Contributors are people who deeply care about making open-source software more inclusive.

For some p5.js contributors like Aarón the emphasis on community is what motivates them to work on the project. "I decided to contribute when I realized that community and ethics were more important than the product itself," they tell me in early October. Similarly, evelyn explains that "contributors care more about the people that use the software and that work on it than the software itself." evelyn's view resonates with plenty of the conversations I've had with contributors. This perspective is different from many ways in which FLOSS is theorized. Usually the social organizations of FLOSS are seen as a medium for writing efficient code; with emphasis on the "bazaar." The writing of code is the focus, not the people who make the bazaar possible. Lehmann argues for this common software-centered view saying that "to (nearly all) FLOSS developers, writing software is the focal point [...] the structure of projects is shaped by this, not the achievement of some social change." However, when I talk to contributors, most emphasize how they care about community, inclusion, and changing the social nature of FLOSS. "We don't want to make the software with the most impressive graphics, it's about people, it's about beginners, that nobody feels left out," a contributor tells me.

The p5.js contributor identity is different from most FLOSS developer identities because the p5 project itself was born with a clear goal of transforming how and who gets to write software. Lauren succinctly describes this: "I originally got involved with this

work because I was frustrated by the lack of diversity in the open source world. I felt alone as a woman of color and wished for a space that was more welcoming to queer, trans, women, and people of color. p5.js became a way for me and many others who felt underrepresented, excluded, or intimidated to create an alternative together." Contributors share a commitment towards inclusion and making participants feel welcome. Cassie is "really impressed by how willing people are to be friendly. Because that is quite uncommon in the open source world. A community being friendly is so radical. And that's weird... why is that so radical?" "I feel welcomed and accepted here, which is something I don't feel in a lot of places," Qianqian says. Xin agrees: "I always feel listened to and respected. It's shockingly a rare experience." In the process of creating a project that prioritizes people and inclusion, contributors engage with critical questions about the nature of software and even what it means to be a contributor.

Contributors recognize that writing code is not the only way to become a contributor; they think teaching, organizing community events, and creating tutorials and documentation are valuable ways of contributing and building community. One contributor that I talked to even suggested that perhaps final users should also be considered contributors because without them there would be no reason for developing or teaching p5.js. However, contributors are aware that currently the community gives more importance to software development over other ways of contributing. Xin and Cassie wonder how contributions that are not code—teaching, community organizing, making projects—can be shared, encouraged, and celebrated in the online community.

In thinking about the contributor identity, and who gets to be a contributor, contributors recognize that there's a lot work to be done to make the project more inclusive. For some it's urgent to increase access for beginners and people with disabilities. evelyn says that talking about inclusion and saying that everyone is welcome is not enough, she asks: "How can we prioritize people who have been marginalized over people who are welcome to any project?" There are some concerns about how the ideas behind the software are presented to final users, a contributor wonders how resources for learning how to code can incorporate questions about the role coding and software play in society. Other contributors recognize that the idea of a contributor carries values that might not easily translate to places with more conservative and authoritarian governments.

These are some of the things I've learned while working on an ongoing larger project about the contributor identity. Like many contributors, at first, I was hesitant to identify as a contributor, and I am still not fully comfortable with that label. But I am grateful to be part of this community and for the conversations the community is willing to have about what it means to be a contributor, what it means to teach or make software, how software should be built, and who gets to participate.



PER 012 Getting Picked Last, 2020, Dorothy R. Santos.

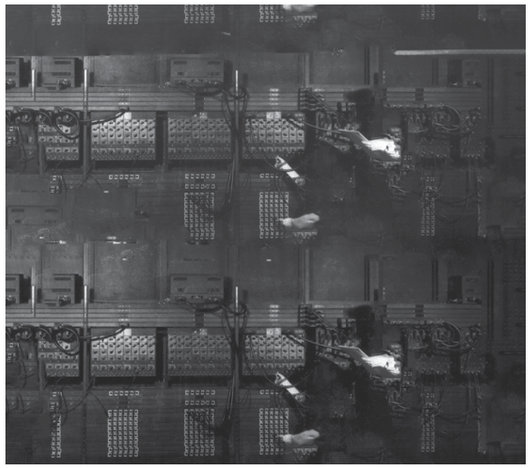


FIG 012-01 Content Aware Collage, Alice Mira Chung. Grayscale photograph showing a grid of computers with a washed-out human form handling the cables, one in each row (four rows total). Based on U.S. Army photo research archives.

All of the big ideas I come up with are always better in my head, such as learning code. Not that I ever thought learning would be easy, but I expected to pick it up rather quickly. The past few years have been humbling when it comes to learning code. Sometimes, I'm embarrassed to say that I'm not an expert coder especially as someone involved in the open-source software for the arts community. I would like to see myself as someone who advocates and uplifts the work being done through creative code. While it has only been a handful of people who have said, "You do so much work with Processing Foundation, I figured you just know how to code really well," it becomes disheartening. I wonder if I belong. I often ask myself, "Who do you think you are?"

The memory that hurts me when I think about it was the time I took an Introduction to Programming for Artists hybrid seminar (my second year as a PhD student) and feeling awful within the first week. I didn't have the same skills as my classmates. I also wondered why on earth people who are so well-versed in programming enrolled in the class, but I let this thinking go. I couldn't let anyone's knowledge and experiences diminish me. I was there to learn. I was excited. I did the homework and worked hard to understand. Yet if I knew that the majority of work entailed watching nothing but online tutorials with little discussion and guidance on coding languages and art, I would not have enrolled!

The last few weeks of the class seemed to sneak up on me and I was paired with someone who knew how to code well (also I wasn't picked or asked by other groups or people to join their team; likewise, I was too scared to ask to join others). The instructor met with us and said to me, "You two will have to figure out how you will work together because I don't want **** to end up having to do all of the coding." My heart sank. I sat on this comment. **** didn't say anything. As a matter of fact, I was disappointed in her for running off and doing her own project without communicating with me at all! I cried from the disappointment. This story might sound silly to someone reading

this who has always been encouraged in the hard sciences or someone who is a masterful coder. And yes, I get that I have skills others don't have. I write, facilitate, encourage, create weird stories, and I can even read through theory and have some relatively interesting conversations with my classmates. I also read tarot cards for people and I seem to have a knack for being a cheerleader for community members. But I yearned to gain some kind of technical skills. I still do.

Even though I'm a lot older and a complete novice, I'm ecstatic when I practice and code works. I'm thrilled and I don't care if hundreds before me figured out how to create a text generator in p5.js. I FIGURED it out by myself. I guess what I'm trying to say is that the open-source community, especially the p5.js community, has been the most supportive and caring when it comes to my goals and aspirations of making art and creating new ways to present my writing.

I ended up doing a project for the class (and passed!) because I watched Dan's tutorial on using p5.js with the riTa library (https://www.youtube.com/watch?v=IIPEvh8HbGQ) created by Daniel Howe (https://rednoise.org/rita/) and was inspired by some of the projects I researched in computational literature. I think one of the biggest takeaways was speaking with my professor and telling her I didn't appreciate her lack of confidence in my abilities. While she was apologetic, I started to wonder and imagine what other students like me might be going through. Perhaps this is why Dan, Lauren, Casey, Ben, Saber, Johanna, and I care so much about the community and believe in accessibility and inclusion. There's so much work to do because I had to go outside the structures of my institution to get the knowledge and skills to do the work I should have been learning from within it. I should have felt supported, but I had to look elsewhere. Thankfully, I didn't have to look too far and everything I needed was within reach. This work is not easy and it takes so much intellectual, emotional, and physical labor to be present. My hope is to continue advocating and uplifting all of the work done within and for the p5.js community because it's been so welcoming and open to me from the start.



PER 013 Imagining a We, 2019,
Xin Xin.

FIG 013–01 PCD@LA 2019!
A group photo of approximately 140 people stand outside on a lawn.

Organizing Processing Community Day, Los Angeles was an incredible experience. As a community organizer and an educator, I often need to imagine a we during a planning process. This leads to a series of questions I ask myself: Will the participants find the program interesting and beneficial? Will my students see the things I see and feel the things I feel? Am I including challenging but important subject matter? Am I proposing a world that is intentional and empowering?

Starting from the first day of organizing PCD@LA, imagining a we led me to ask this question: Are we interested in co-creating a time and space where artists, designers, coders, activists, children, and teenagers don't need to pretend that our realities are mutually exclusive? And, can we engage with one another person to person, rather than professional to professional?


FIG 013–02 A.M. Darke, track coordinator, introduces the Radical Pedagogy track.
An audience of roughly 40 people are seated in an auditorium. A woman speaks from a lectern. On the right is a projection on a large wall, with blue background and white text that reads: "Radical Pedagogy, A.M. Darke – Dorothy Santos." On the left is a smaller screen with CART captioning, in white text on a black background. The text reads: "So there are so many people that do emotional labor and do work within social justice circles and within open-source communities who are just saying, this is what I learned."

These initial questions became a guiding principle during the concept phase, open calls, visual promotions, to designing and implementing the program. After working with the organizing team through an eight-month planning process, we were able to gradually implement these underlying questions in various aspects of the event. What we ended up with was a total of 65 conference guests presenting in four differently themed tracks, with self-organized sessions, live-captioning, childcare, and individual communication channels dedicated to POC and queer folks.

The size of the program was ambitious but highly intentional. Without this

FIG 013-03 Participants of the workshop "My Face is My Own! : Helping Kids Beat Facial Recognition Software with README," from the Epic Play track.

Three people are pictured. In the foreground, a child paints blue lines on the forehead of an adult. In the background, an adult wearing a mask that resembles flower petals takes a selfie.

we wouldn't have been able to bring a multitude of people into the same room and include a wide array of important topics relating to art, technology, and activism. For this I am extremely proud and grateful for all the lightning talk speakers and session organizers whose contributions truly gave the day its substance.

Making an expansive and inclusive program with a small team means that there were many moving parts to oversee throughout the day. Of course this created a healthy dose of anxiety prior to the event. For months I played out many different scenarios of how each part of the day could happen. Many reflections and doubts were related to whether I was making the right decisions and framing the event in a way that would produce empowering outcomes. But as we started to count down to the event, these anxieties became unproductive, and I realized that after all the organizing effort, it was left to the PCD@LA community—which wouldn't be fully formed until the day of the event—to create the Processing Community Day they'd like to have.

In order to create something new we also have to not be afraid to fail. Failures are so scrutinized in the society we live in, though they are often accepted as part of everyday life in certain non-Western cultures. There are a couple things that took place during the organizing process and the event that I would do differently in the future. For instance, designing a different presentation duration; providing higher compensation to art collectives; and reserving budget to hire an additional person to commit to community outreach, specifically to youth and kids' programs, and the disability community. Prior to posting the open call, I would have loved to have gotten more feedback from these communities on what they would like to see from the programs.



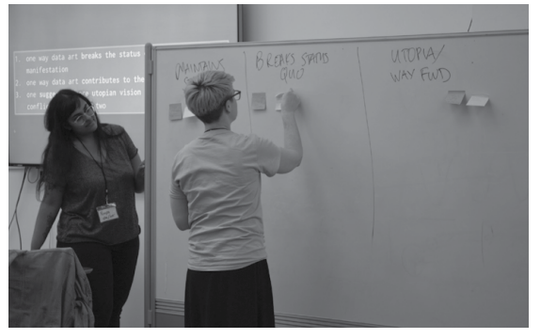



FIG 013-04 From "Toward a Reflexive Data Culture," a workshop with Roopa Vasudevan, part of the track, Under the Silicon, the Beach!

Two people stand in a classroom. One watches the other put Post-Its on a whiteboard. The whiteboard has been divided into three columns, which read: "Maintains Status Quo, Breaks Status Quo, and Utopia/Way Fwd." Behind them is a projection on a wall of a slide with text that is partially covered by the whiteboard. Some text that can be seen reads, "1. one way data art breaks the status...manifestation, 2. one way data art contributes to the... 3. one... utopian vision..."

I feel very lucky that the event was extremely well received by the many people I came across during the day. I have never heard anyone describe a conference with the word "magical," but this word was brought up to describe PCD@LA by a couple people I spoke to. This was such a pleasant surprise, and not something I could have anticipated prior to the event. It can only happen at the moment and for the moment, through the co-creation of conference guests, participants, organizers, and volunteers.

PER 014

# p5.js Editor, A Personal History, 2021,

Cassie Tarakajian.

I began work on the p5.js Editor over five years ago. I was given the opportunity when Dan Shiffman and Lauren Lee McCarthy approached me with some funding from the Interactive Telecommunications Program (ITP) at NYU to start the project. Despite applying for but not receiving a Processing Foundation fellowship, I was zealous about being involved with the organization. I worked in tech as a software developer for a few years, but this was the first software project I led. It was also the first open-source project I contributed to. I was excited to work with folks I admired, who seemed to trust I would do a good job, though I didn't really understand why. I had a simple version of the Editor working within a few months of starting, which made teaching p5.js easier. As the project matured, I started teaching, mentoring, designing, doing user research, and project management. By the fall of 2018, I publicly announced the project, when I felt like it was stable enough. A year later, one million p5.js sketches had been created using the Editor. Now, five years after starting this project, growth has accelerated, with over five million sketches created, and there are no signs of the project and the work slowing down.

For the Editor to have grown exponentially in the past few years, I wanted to take this moment to record its history. As I sat down to create a timeline of events, I quickly realized the shortcomings of this process. At the time of writing this, 805,891 people have created accounts on the editor. That means there are at least 805,891 separate histories of the p5.js Editor. However, I have only experienced one of them. Rather than seeking to document a comprehensive history of the p5.js Editor, instead, I want to share my experience.

This project is much bigger than just me. According to GitHub, 121 different people have contributed code (https://github.com/processing/p5.js-web-editor/graphs/contributors) from around the world. This number doesn't begin to capture those who have created issues, taught a workshop with the Editor, written their first line of p5.js code, or even forked their own version. It also doesn't capture the people who have supported me when I've felt down about the project or those who have helped me celebrate milestones. As a community, we have accomplished this together. While this is my story, my story has been impacted by all of you.

A Timeline of the p5.js Editor

2014 SUM Sam Lavigne starts the p5.js Desktop Editor as a Google Summer of Code (GSoC) project.

2015 SUM Sam Lavigne mentors Guy de Bree by porting p5.js Desktop Editor to Windows as another GSoC project.

SUM Jason Sigal creates a Web IDE for p5.js, mentored by Dan Shiffman, as a GSoC project.

2016 MAY Cassie Tarakajian begins working on the p5.js Web Editor, with support from NYU ITP.

JUN The first workshop using the p5.js Web Editor is taught by Cassie Tarakajian at ITP Camp.

SUM Mathura Govindarajan works on web accessibility features in the p5.js Web Editor, mentored by Claire Kearny-Volpe.

AUG The second workshop using the p5.js Web Editor is co-taught by Cassie Tarakajian and Emily Xie at Pioneer Works.

FALL The p5.js Web Editor is tested at NYU ITP, UCLA DMA, and other university programs.

FALL Yining Shi adds new features to the p5.js Web Editor: detecting and preventing crashes from loops and importing the p5.js examples.

2017 JAN The p5.js Desktop Editor is deprecated in favor of the p5.js Web Editor.

SPR Cassie Tarakajian is a Processing Foundation fellow, continuing work on the p5.js Web Editor. See their post, A p5.js Web Editor for All (https://medium.com/processing-foundation/a-p5-js-web-editor-for-all-64aaa3f9d767).

SPR Andrew Nicolaou is a Processing Foundation fellow, working on new features for the p5.js Web Editor. See their post, Features and Fixes in the p5.js Editor (https://medium.com/processing-foundation/features-and-fixes-in-the-p5-js-editor-722e4b56495e).

SUM Jen Kagan works on code autocomplete features and Zach Rispoli works on classroom features, as GSoC projects mentored by Cassie Tarakajian.

SUM NYC Department of Education begins using the p5.js Web Editor as part of CS for All curriculum, with support by Luisa Pereira, Jose Olivares, and Aankit Patel.

SUM Jerel Johnson creates new UI designs.

OCT Processing Community Day takes place at the MIT Media Lab. Cassie Tarakajian makes their first presentation about the p5.js Web Editor to the Processing community.

2018 MAR NYC Department of Education partners with Processing Foundation to support development by Cassie Tarakajian, with support from Ana Giraldo-Wingler.

SPR Mathura Govindarajan and Luis Morales-Navarro move accessibility features in p5.js Web Editor to separate library, p5.accessibility.js, as a Fellowship project.

SUM Liang Tang makes updates the console, as a GSoC project mentored by Cassie Tarakajian.

SEPT The official public release of the p5.js Web Editor. The URL changes from alpha.editor.p5js.org to editor.p5js.org. See the post Hello p5.js Web Editor! (https://medium.com/processing-foundation/hello-p5-js-web-editor-b90b902b74cf)

SEPT Dan Shiffman begins using the p5.js Web Editor on his YouTube channel, Coding Train.

FALL Cassie Tarakajian teaches their first full semester class, Introduction to Computational Media, using the p5.js Web Editor.

OCT Generative Design releases a second edition, with examples written in p5.js, and ported to the p5.js Web Editor by Joey Lee. This is the first published reference of the p5.js Web Editor.

NOV Jerel Johnson makes updates to the colors and creates a design system.

2019 SUM As a GSoC project, Rachel Lim adds the ability to search sketches, mentored by Cassie Tarakajian.

SUM Andrew Nicolaou adds the ability to create collections, the Dashboard view, and a public API.

AUG At the p5.js Contributors Conference, Cassie Tarakajian and Luca Damasco draft a new description of the p5.js Web Editor, and begin referring to the project as the p5.js Editor.

FALL Cassie Tarakajian teaches Introduction to Computational Media for a second time, at NYU ITP.

DEC The p5.js Editor surpasses one million sketches created.

2020 SPR Khensu-Ra Love El implements privacy settings for sketches.

MAY p5.js Editor starts using versioning, with the v1.0.0 release.

SPR With a grant from the Clinic for Open Source Arts (COSA), Cassie makes web accessibility updates to the p5.js Editor, such as adding ARIA labels and adjusting color contrasts. See their post, Accessibility Improvements for the p5.js Web Editor (https://medium.com/processing-foundation/accessibility-improvements-for-the-p5-js-web-editor-3d76b7c68c6d).

AUG Cassie Tarakajian becomes p5.js Editor Lead at Processing Foundation.

SUM Ghales Trilo adds mobile design, mentored by Cassie Tarakajian, and Omar Verduga adds Spanish translation, mentored by Andrew Nicolaou.

OCT The p5.js Editor surpasses three million sketches created.
NOV Processing Foundation receives a grant from Grant for the Web to develop a subscription model for the p5.js Editor.

2021 SPR Connie Ye adds new tests and a testing guide, mentored by Cassie Tarakajian with support from the STUDIO for Creative Inquiry.
SPR New Media Rights drafts a Terms of Use and Privacy Policy.
SEPT The p5.js Editor surpasses five million sketches created.
SEPT What's New with p5.js and the p5.js Editor (https://medium.com/processing-foundation/whats-new-with-p5-js-and-the-p5-js-editor-2045f5ca774e) is published, sharing new p5.js Editor features: many new translations, an interactive console, and debugging improvements.



PER 015 Notes on Activism, 2020,
Johanna Hedva.

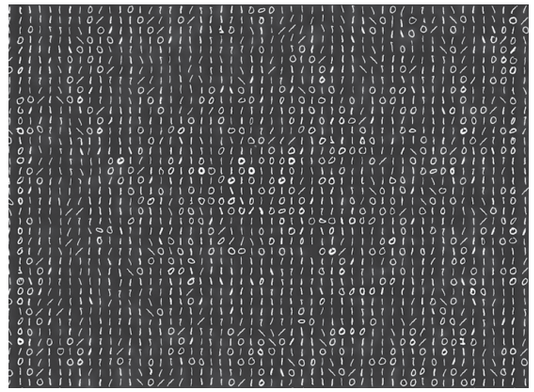


FIG 015-01 Content Aware Collage, Alice Mira Chung. An assembly of white zeros, ones and dashes positioned in a grid covering the entire gray canvas. This is a computer-generated illustration based on zeros and ones from MNIST database.

The thing about activism is that it's hard. It fails far more often than it succeeds. It's exhausting. It's frustrating. It burns you out. Any activist can tell you this but, for various reasons, it's not much talked about. Indeed, it's minimized, ignored, and even repressed. At the monthly meet-up with your collective, you're not supposed to spend your time talking about how, say, the organizers got in an argument because one of them said something problematic, or how the website broke so no one could donate, or how only four people came to the last workshop, or—the list goes on. One ought not, seems to be the official rule, give too much space to this—the messy reality of activism—because it sits uncomfortably with one's aims. When you're trying to make the world not only a better place, but an entirely new one, being depleted, discouraged, and dismayed seems wrong. Where does fatigue fit into insurrection? It's so incongruous to your ideals because it reminds you how the world *actually* is, not how you would like it to be. And this is a difficult truth to bear because it emphasizes how much more work there is to do.

I've worked in activism for most of my adult life, and I've been getting paid to do it since 2014 by the Processing Foundation. But I don't call myself an activist. Most activists I know don't call themselves such. And we all tend to give the same reasons why: we say that there are so many others who are doing "real" activism, that our pitiful attempts don't warrant the weight of the title. I bet this is not the actual reason. I think what we really mean is that activism is too arduous, too full of failure and defeat, for it to be given the vaunted status of a name that shares its root with the word "action." So much of activism feels like the opposite of action: it's a lot of little labors that are invisible but necessary; it's so many delays, so much waiting; it's things not working out the way you'd hoped; and it's coming to terms with why your attempts failed, and may always fail. It's persisting in the face of this—that you may *always* fail—and still choosing to try again.

The word "success" has been around for millennia, and its meaning has changed throughout history. In its earliest inception, it was the combination of two Latin roots, "sub" and "cedere," which meant "next to, after," and "go, move," respectively. Success,

as in “to succeed,” originally meant what came after, what followed, and it was a neutral fact, neither good nor bad. It had nothing to do with triumph or achievement, and didn’t come to mean what it does now—the accomplishment of a desired result—until the 1580s. It’s not a coincidence that that development occurred then. Capitalism began to take root in Europe in the 16th century, so it makes sense that how *well* a thing performed would come to measure the thing’s *value*.

This capitalist tenet is one that we’d do well to leave behind, because it’s not actually how the world works. Although it structures our reality, it’s not very realistic in practice. Most things don’t perform well: they break, don’t work, take longer, are expensive, and need more support than you thought. And when I say “things” I mean machines and objects, but also bodies, relationships, ideologies.

When it comes to “success,” I’ve come to learn that the more exhausted you are, the more successful your activism has been. This is because, at its heart, activism is about trying to do something that’s never been done before—and the most important part of that sentence is the word “trying.” Exhaustion is the sign that you tried, that you gave it all you had. It’s not a done deal, it’s only ever an attempt.

It’s not just that there are no role models or guides. It’s not just that the world and its institutions are structured in opposition to what your activism is trying to do. It’s that, *because* activism is about radically re-doing how we do things, and have done them for generations, it will necessarily fail far more than otherwise, because no one has ever *done* this before. And not only is it new, but it agitates against a world that does not want to change. It’s trying to do something that those in power don’t want to do. Of course it’s gonna fail.

Maybe it’s a platitude to say that, but it gives me comfort. I’m reminded that activism shares its root with another word—*actual*. As much as activism is about dreaming into being something that doesn’t exist yet, it’s also about standing face-to-face with the actual mess of life. In that mess, there is exhaustion and failure, sure, because that’s what life is made of.

Failure has become a thing that successful people in successful fields like to talk about. In those fields, which champion success as capitalism defines it, failure is positioned as valuable because it teaches you something, which pushes you to be “better.” I like the idea of failure being neutral, neither good nor bad, just a fact of life, what comes after what is done. I like thinking that activism *will always* fail. Because it means that the decision to take action, even in the face of certain defeat, is pure. It’s not going to get you paid, or bring you victory. It’s trying to do something that has never been done before, in the face of an indifferent and hostile world that doesn’t value this kind of action. It’s important because it is important. It is persisting. And that is enough.

The Processing Foundation Fellowships support artists, coders, and collectives in visionary projects that conceive a new direction for what Processing as a software and a community can do. Fellowships are an integral part of the Processing Foundation's work toward developing tools of empowerment and access at the convergence of art and technology. Fellowships emphasize projects that expand Processing and its affiliated projects, as well as the evolution of a Fellow's practice. Work done by Fellows is supported through funding and mentorship from The Processing Foundation.

FEL 001

# Public, Private, Secret—Johanna Hedva with Charlotte Cotton, 2016, Johanna Hedva and Charlotte Cotton.

FIG 001-01 2019 p5.js Contributors Conference at Frank Ratchye STUDIO for Creative Inquiry at CMU. Twenty people sit in small groups working together in a large meeting room.

Johanna Hedva and Charlotte Cotton discuss events curated in conjunction with Public, Private, Secret—on view at the ICP Museum (250 Bowery) from June 23, 2016 to January 8, 2017.

Jonanna Hedva, Director of Initiatives at Processing Foundation.
Charlotte Cotton, Curator-in-Residence at Processing Foundation.

CC Johanna, it is a real pleasure to have collaborated with you on a series of events with the Processing Foundation Fellows. I'd like to start by asking you to describe the recently formed Processing Foundation, where you are Director of Initiatives, and its formation and aims.

JH Thank you, Charlotte! The Processing Foundation was founded in 2012, more than ten years after the original Processing software was invented. The Foundation is the nonprofit organization that manages the development of Processing, and now p5.js and Processing.py. These three software projects are open source (free, non-proprietary, and community-made) and created by artists, for artists. The Foundation's aim is to nurture the communities that make and use them. We have a Fellowship Program, which funds creative and technical research by artists, coders, and collectives; an Education Program, which develops curriculum and resources for educators in art and computer science departments; and an Initiatives Program, which I've been leading since 2014. The Initiatives Program is an effort toward making our projects accessible to diverse groups, and addressing issues around access, like race, gender, sexuality, class, and ability. We do this by partnering with organizations—like the International Center of Photography!—to make community events that emphasize dialogue around social and political issues in the fields of art and technology.

CC And the mission of the Processing Foundation is a primary reason why we wanted you to curate a series of events at the ICP Museum. Can you tell me about your motivations for setting up a Fellowship program at the Foundation?

JH The Fellowship Program began informally in 2013 with Greg Borenstein, Lauren McCarthy, and Wilm Thoben, who were already active members of the

Processing community. They were offered fellowships as a first iteration of the idea to fund new work that was part of their own practices and also aligned with projects the Foundation was interested in. This first round was informal in its timeline and requirements, and primarily led by the fellows' own interests. For instance, Lauren McCarthy was interested in making a Processing library for JavaScript, which would require her to learn how to make it as she went along. Her fellowship was extended after a great finish on the first fellowship, and now is a thriving project—p5.js—that the Foundation continues to develop with her. Greg Borenstein and Wilm Thoben were hired to complete work on libraries that they were already building for their own practices, and which have been included in new releases of the Processing software.

These projects influenced how the Fellowship Program's first open call, in 2015, was structured. For the 2016 Fellowships, we were looking for projects that ranged from exploratory research and software development, to community outreach initiatives that specifically addressed barriers around access. A fellow's technical expertise was a lower priority than was their interest and enthusiasm in exploring a new topic and proposing a project that would expand what the Processing community can encompass. A fellow receives close mentorship and a stipend of $3,000, at $30/hour for a total of 100 hours, which can be completed anytime in a six-month period.

CC The structure of your Fellowship Program is really interesting on a number of levels, not least the clarity of the labor and its remuneration. I don't think that I have seen this clarified to such a degree within the realm of artists' fellowships. I also wondered if you could also say more about the ethos of your Fellowship Program and specifically what you are suggesting for creative relationships with technology, what it says about technical expertise at this juncture of social history. I may be projecting here but it makes me think about the role of artists in technology and the breaking down of barriers to access.

JH I love this question! Yes, the inclusion of how the fellows' labor is paid for, and the specifics—or lack thereof—around when and where it can be performed, are directly related to how both open-source projects and artistic practices tend to function. Open-source projects are, by definition, entirely community made, maintained, and operated. They are based on volunteer systems, in that they require a lot of devotion and free and unpaid time. You might have a day job, and work on open-source on the weekends or at night; you might have some time between freelance work where you devote every waking hour to the project. But you do so because you are emotionally invested and you are part of a community who is too. This is, of course, also a lot like artistic practices: the precarity of them, the necessity that you be so devoted you are willing to forgo money for your labor, but also the inspiration, enthusiasm, and relationships that keep you committed. Our fellowship program wants to highlight the good parts of this model—the passion and devotion, the community involvement—while trying to mitigate the bad—the unpaid labor.

You're right that our program is based on an idea of how technology can be both a viable artistic medium as well as a model for sociality. The Adobe model—that software for creative work is expensive and proprietary, and how it's built is kept hidden from those who use it—represents everything we are trying not to do. This comes from a belief that software is a medium and a tool, and learning to program can be a creative, exploratory practice.

We are working with the question: What happens when artists, individually and as a collective, make their own tools? This is less about having technical expertise, per se, and more about wanting to participate in the process of building something, and learning the technical skills as a result. It also points to the fact that a community is a helpful and perhaps necessary component to the development of a new medium— as the history of photography as a medium also reveals.

CC I suspect I have responded so much to the work that you are doing at the Processing Foundation in part because it's such a shining example (and a parallel to photography) of perceiving software as a medium and then deeply considering how this new medium could be a tool for social change. Can you tell me about the kinds of projects that the Processing Fellows are developing with you?

JH We are very excited about our 2016 Fellows, as this was the first time we held an open call for applications. For 2016, we awarded five fellowships—our most to date.

Allison Parrish is a poet who writes with code, or, you could say, a computer programmer who writes poetry. Her fellowship has been to develop a version of Processing that is written in the programming language Python.

Claire Kearney-Volpe's fellowship work extends her research at the Ability Lab at NYU, which has advanced our software toward being accessible for blind and visually impaired people. It's an important question to consider what "visual programming" and "visual art" are, and how they can be newly conceived, from this perspective.

The Digital Citizens' Lab, a design collective that focuses on civic technology, created tools for educators that can meet the needs of historically underserved children of color. They have been working on an interactive game, illustrated with comics, whose narrative can be re-written as the game is played. Jess Klein and Atul Varma, who are designers interested in "building bridges between humans and machines," have been working on accessibility in p5.js, in terms of human-centered usability.

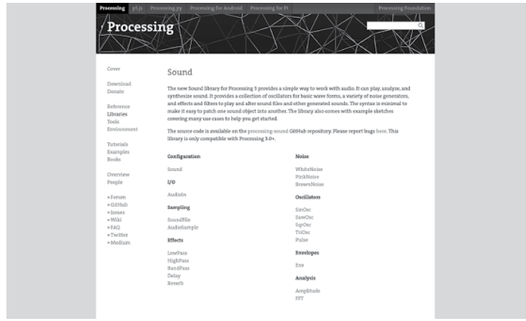
Luisa Pereira and Tega Brain both teach at the intersections of computer science, engineering, and art. Their fellowship work has been to develop p5.js as a tool for teachers working in interdisciplinary fields, and to develop resources, discourse, and a sense of community in this burgeoning field.

CC It's a great honor for ICP to host three of the Processing Foundation Fellows' projects as part of Public, Private, Secret's public programming. Can you tell me about the wider picture of public engagement that you have planned and what its aims are?

JH Likewise! It's been fantastic to work with you and ICP! Since its founding in 2001, Processing has enjoyed and relied on a robust community. One of the goals of starting the Foundation in 2012 has been to expand that community to groups who might not yet have felt invited into it. This feeling can happen for many reasons, from lack of resources, to impenetrable clique-ish vibes, to micro-aggressions, etc. In the fields of art and tech, the word "diversity" is often deployed these days as a business mandate, as something that will be better for the company first, its users second. We're striving toward a kind of diversity that is cultivated from a grassroots model: an effort toward diversity that is sensitive to why there's been a lack of it in the first place, and encourages the nuanced process of building something different. "Diversity" has little consequence if it is not responsive and supportive to the specificities of different groups. The aims for our public engagement programs are very much informed by this, and so strive to be human-scaled, collaborative, and shaped by discussion and interdependency within and between different communities.



FEL 002 Processing Sound Library, 2013, Wilm Thoben.




A screenshot of the Processing site showing the article titled "Sound."

Wilm Thoben developed a new core Sound library from fall 2013 through winter 2014. This library was also released with Processing 3.0, providing a simple way to work with audio. It can play, analyze, and synthesize sound. The library comes with a collection of oscillators for basic wave forms, a variety of noise generators, and effects and filters to alter sound files and other generated sounds. The syntax is minimal to make it easy for beginners who want a straightforward way to add some sound to their Processing sketches!

FEL 003 p5.js, 2013, Lauren Lee McCarthy.


White text that reads "p5*" on a hot pink, square background.

Lauren Lee McCarthy started the work that has now become p5.js in spring, summer, and fall 2013. p5.js is a JavaScript library for creative coding, with a focus on making coding accessible and inclusive for artists, designers, educators, beginners, and anyone else! p5.js is free and open-source because we believe software, and the tools to learn it, should be accessible to everyone.

Using the metaphor of a sketch, p5.js has a full set of drawing functionality using the HTML 5 canvas element. You're not limited to the drawing canvas though. You can think of your whole browser page as your sketch, including HTML 5 objects for text, input, video, webcam, and sound.

p5.js draws inspiration, wisdom, and guidance from its precursor Processing. However, it is a new interpretation, not an emulation or port. We don't aim to match Processing's set of functionality exactly, allowing ourselves space to deviate and grow in the context of the web.

FEL 004 OpenCV for Processing, 2013, Greg Borenstein.

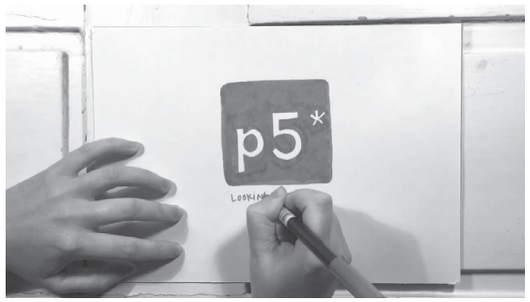
Greg Borenstein expanded and released the OpenCV library for Processing in spring and summer 2013. The library is based on OpenCV's official Java bindings. It attempts to provide convenient wrappers for common OpenCV functions that are friendly to beginners and feel familiar to the Processing environment.

A screenshot of four frames of the same face but with different contrast and brightness levels in a pop-up window titled "BrightnessContrast." The top-left frame is in color with low contrast and high brightness, the top-right is in color with low contrast and brightness, the bottom-left is in gray tones with high contrast and low brightness, and the bottom-right is in gray tones with high contrast and brightness.

FEL 005 Learning p5.js, 2016, Tega Brain and Luisa Pereira.

Working with a number of collabortators, Tega Brain and Luisa Pereira ported many tutorials from Processing and created some new ones, too. They also revamped the p5 tutorials page, adding new resources and organizing the information to be more easily navigable.

One of the most exciting additions was the creation of a video tutorial that provides a beginner-level introduction to developing the p5.js library. It is a goal with p5.js that anyone who is interested is able to help build the platform. However, all of the files and conventions of the p5.js development environment can be confusing for someone new to the project. Luisa's illustrated video makes no assumptions about what someone might already know, explaining the file structure and development tools and processes clearly and concisely. There's also a version in Spanish!




A hand drawing of the p5.js logo, with someone's hands in the process of writing "Looking" underneath it.

Their work concluded with a Learning to Teach panel led by Tega at the International Center of Photography. The panel, with educators De Angela Duff (New York University), Aankit Patel (NYC Department of Education), and David Sheinkopf (Pioneer Works), focused on pedagogy and the creation of environments and tools for learning interdisciplinary computational arts practices in universities, high schools, and creative communities.

FEL 006 Friendly Errors, 2016, Jess Klein and Atul Varma.

Jess Klein and Atul Varma had the goal of "making p5.js the most helpful JavaScript library in the world" by improving its friendly error system through the lens of human-centered design. They began by observing the most common errors beginners encounter through learning and teaching p5 themselves, and observing others doing so.

Jess, new to p5.js, began by reading the documentation materials and trying to follow the tutorials to create her own sketches. As she worked, she identified the confusing parts and pain points of the platform and learning materials. Jess's work shows that you need not be an expert programmer to contribute to development, and that actually a beginner can offer insights that someone more familiar with the system does not have.

The next step of the project was to use their insights to devise new ways of making the p5.js library even more helpful. Atul and Jess made a number of improvements to p5.js error reporting, trying to catch common errors and provide more information about how to resolve them.

For example, with JavaScript, it is easy to overwrite existing variable or function names without knowing it. This would cause a previously defined variable or function to fail or work incorrectly, causing much confusion. To remedy this, they added functionality that would notice if the user attempts to overwrite a p5.js variable or function, and warn them of what was happening.

For example, the following sketch:

```
function setup() {
  text += "blarg";
}
```

logs this warning:

```
You just changed the value of "text", which was a p5 function.
This could cause problems later if you're not careful.
```

Hopefully this change will make it easier for users to debug their sketches.

Documentation of new p5.js friendly errors.

In observing teaching sessions, Atul noticed that demonstrating and sharing simple examples in class was often unwieldy – they were hard to format, and often required students copying and pasting code from a website into their own editors to see the example in action. In response, he created p5.js widget, a reusable tool that makes it easy to embed editable p5.js sketches in blog posts, interactive curricula, and other places.

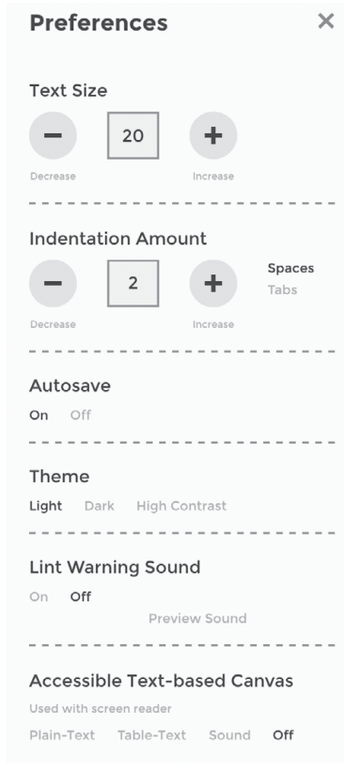
FEL 007 The Processing Accessibility Project, 2016, Claire Kearney-Volpe.

The goal of Claire Kearney-Volpe's work was to make the forthcoming p5.js editor, learning modules, and web content usable by people who are blind or visually impaired. Working with an advisory panel of users and experts in the field, Claire researched, designed, developed, and tested prototypes of interfaces for the p5.js editor and learning resources. The aim was to make them accessible across a variety of browsers, operating systems and assistive technologies.

Some of the most interesting and experimental work involved the output of the editor, originally an HTML canvas that was impenetrable by screen readers (a software application that converts text on a page into audio speech). Claire, along with Mathura Govindarajan and Cassie Tarakjian, worked on creating a "Shadow DOM" that dynamically interprets the visual and spatial outputs on screen and renders them into a text description.

The preferences panel of the editor allows the user to choose from various types of feedback based on what type of sketch they are creating. For example, if they are making a static drawing, they can choose to have the canvas described in plain text. If they are creating an animation, they can choose to be alerted when objects pass into particular areas of the canvas. Stay tuned for the release of the p5.js web editor in 2017, which will include all of these features!






The preferences panel that includes accessibility options for the p5.js web editor. The different settings read "Text Size," "Indentation Amount," "Autosave," "Theme," "Lint Warning Sound," and "Accessible Text-based Canvas."

The fellows ended up collaborating in ways that drew on each of their individual projects. In the Signing Coders workshop series led by p5.js collaborator Taeyoon Choi, Luisa Pereira, Atul Varma, and Claire Kearney-Volpe joined forces to teach p5.js to who that are deaf or hard of hearing. They experimented with a variety of custom learning tools and activities, iteratively developed over the series.

FEL 008 Coding Comic, 2016, Digital Citizens Lab (Evan Wu, Sharon Lee De La Cruz, and Leslie Martinez).

The Digital Citizens Lab, consisting of Evan Wu, Sharon Lee De La Cruz, and Leslie Martinez, sought to use p5.js as a tool for outreach and education, specifically designed for children of immigrants and people of color. They describe the goals of their Coding Comic project as follows:

"The ed-tech industry does not consider the education inequity that urban children of color face. The industry assumes that there is a standard of education that all Americans are receiving and then they build products based on this assumption. Coding Comic is a tool that emphasizes the logic behind coding instead of actual coding languages. Our goal is to


A scene from the Coding Comic web application consisting of three panels. The top panel contains a character making spooky hands with a speech bubble that reads "El Cuco." The bottom panel contains three children in bed with scared expressions. The right panel contains a text box titled "Narrator" and reads: "The story scared the kids and before they knew it, it was time to go to sleep."

create digital tools and resources for educators that can meet the needs of historically underserved children of color."

They developed a platform where children can rewrite and re-visualize a narrative by changing the logic of events, thus learning about coding in an implicit and intuitive way. When they understood the logic of the story and placed the correct elements or "variables" in the frame, they were rewarded with animations which reinforced their understanding.

The Digital Citizens Lab recently presented their Coding Comic project and held a panel to discuss the challenges and working strategies for STEM teaching in schools with invited educators at the International Center of Photography.

FEL 009 Processing.py, 2016, Allison Parrish.

Allison Parrish's Fellowship proposal included a wide range of improvements for the Processing.py project. Processing was initially released with a Java-based syntax, and with a language of graphical primitives that took inspiration from OpenGL, Postscript, Design By Numbers, and other sources. With the gradual addition of alternative programming interfaces – including JavaScript and Ruby – it has become increasingly clear that Processing isn't a single language, but rather, an arts-oriented approach to learning, teaching, and making things with code. Jonathan Feinberg started Processing.py to extend the ideas behind Processing into Python. Contributors, including Allison, have

been instrumental to its current iteration as a Mode for Processing 3.

Allison's fellowship work was highly technical, but also focused on education. In addition to fixing a number of issues with Processing.py, she created two new Processing.py tutorials, "Processing.py on the Command Line" which shows how to break free from the Processing IDE to develop Processing.py sketches on the command line, and "Python, Jython and Java," which covers how Processing.py brings together Python, Jython, and Java.

FEL 010 ¡Manos a la obra! Empecemos. (Creative Coding in p5.js), 2017, DIY Girls (Sylvia Aguiñaga and Vanessa Landes), Mentored by Lauren McCarthy and Jesse Cahn-Thompson.


Zine cover design by Nisa Karnsomport.

A book cover with blue, red, and yellow shapes and gradients. The title, "Creative Coding Camp," runs around the border of the book.

DIY Girls seeks to increase girls' interest and success in STEAM through new educational experiences and mentor relationships. Their project involved a p5.js (6–8 grade) curriculum development and translation of the p5.js zine into Spanish. Sylvia Aguiñaga and Vanessa Landes expanded, iterated, and improved DIY Girls' current Processing curriculum. They planned week-long creative coding camps for girls and used their feedback to further iterate and improve the material. The creation and implementation of their p5.js curriculum and zine broadened the scope of impact with girls in underserved areas of Los Angeles and in Mexico. Their bilingual zine was a standalone resource for girls to explore – a tool that encourages self-led learning and awakens a love for code.

FEL 011 Processing on ARM, 2017, Gottfried Haider, Mentored by Ben Fry.

Devices such as the Raspberry Pi and the C.H.I.P. are promising vehicles for using code in environments such as teaching and installation art. In the context of everyday life, their ability to execute custom code sketches will also lend itself to ways of actively reflecting upon an area that is increasingly permeated by rather opaque technology, such as Amazon Echo or Google Home.

Gottfried Haider continued to build out Processing's ability to run sketches on those small, ARM-based computers. With many pieces of the puzzle already in place, such as the ability to talk to external hardware (Hardware I/O library) or accelerated video-decoding (GL Video library), emphasis was placed on making it easier to get started and to develop the tools and libraries needed for students, designers, and hobbyists to engage with the aforementioned areas of interest.

Another goal of the fellowship was to advocate for an understanding of Free (Libre) Open-Source Software (FLOSS) culture in design and the arts. With the software stack that, for example, the Raspberry Pi runs on – i.e. GNU/Linux – being developed in a cooperative manner, Gottfried believes that it's important for young designers and developers to be able to navigate this space and to engage with these communities in informed ways.

FEL 012 Everyone Can Code: A Creative Coding Curriculum for Students with Low Computer Literacy, 2017, Niklas Peters, Mentored by Daniel Shiffman.

Through the Processing Fellowship, Niklas Peterssought to make coding more accessible for people with low computer literacy. Learners from marginalized groups often lack a basic familiarity with computers; even if they become more comfortable using computers, coding can seem foreign and inaccessible. By developing a curriculum that builds general computer skills and demonstrates the power (and fun!) of coding from the start of a learner's computing experience,

Programming a Raspberry Pi remotely (using the "UploadToPi" tool).

The Raspberry Pi in focus with a blurred view of a laptop.




Five South African high school students sitting at computers getting familiar with the mouse and menus by using MS Paint.

participants saw coding as a way to express themselves and as a natural extension of using a computer. The curriculum was piloted with high school students in Johannesburg, South Africa.

FEL 013 Community and Code, 2017, Saskia Freeke, Mentored by Phoenix Perry and Johanna Hedva.

Women participating in a workshop and sitting at their laptops

In-person learning can be highly valuable to create a sense of community and to lower the barrier of learning a new tool. To encourage women to explore coding as a tool to express themselves in creative ways, Saskia Freeke developed, organized, and created a series of workshops to teach p5.js in person to women, non-binary, and femme-identifying people in London and Europe, in collaboration with Code Liberation. They worked together with only one simple mission: Teach women, non-binary, femme, and girl-identifying people to program using creativity as a pedagogical approach. Workshops in creation of digital games and creative technologies, game jams, game night, and social networking events offered an informal setting for networking and community bonding to encourage participation.

The workshops are focused on creating interactive playful artworks in p5.js and encouraging women to explore coding to express themselves in creative ways. The workshops will establish a firm introduction to the library and programming basics.

FEL 014 Creative Coding with p5.js for Prisons in Washington State, 2017, Susan Evans, Mentored by Rhazes Spell.

Volunteers sharing and debriefing after a day of volunteering. They travelled from the greater Seattle area to Clallam Bay to inspire and mentor students.

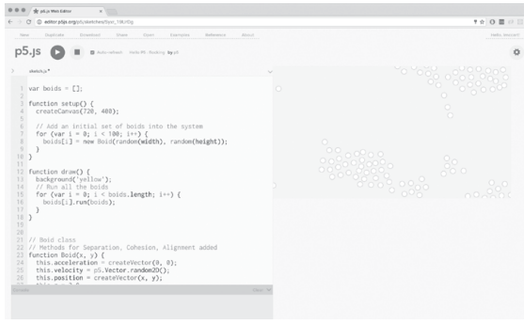
A group of people sitting on a truck smiling.

People in prison deserve open access to education, especially computer science education, which opens access to high-paying jobs. Susan Evans crafted and offered a series of classes in Washington State prisons using p5.js. Since people in prison have limited access to computers and no internet access, Susan developed a paper curriculum to ensure learning could happen beyond the classroom. Assignments and activities engaged and supported students on their path to learning to code.

FEL 015 A p5.js Web Editor for 2017, 2017, Cassie Tarakajian, Mentored by Daniel Shiffman and Lauren McCarthy.

The p5.js Web Editor is an in-browser interactive development environment for writing p5.js sketches. It aims to lower the bar for creative coding. Users can start writing p5.js by simply opening a browser window, without the need to download software or do any configuration. This makes it a great entry point for beginners to p5.js. It does this while maintaining a full set of features, like working with media files and other JavaScript libraries. When users become more advanced, the editor makes it easy to export their work to more complex tools. Because web editor sketches live online, they are more shareable than local files –

which makes it easier for users to showcase their work and get help from the online community. For these reasons, the editor is also a great teaching tool. It is free to use and it is an open-source project.



A screenshot of the p5.js web editor displaying code and the respective sketch.

FEL 016 Features and Fixes in the p5.js Editor, 2017, Andrew Nicolaou.

Andrew worked on the p5.js Web Editor. He worked on general enhancements and bug fixes to the existing Editor. He also explored ways to make the feedback loop between writing code and seeing the possible outcomes as short as possible.

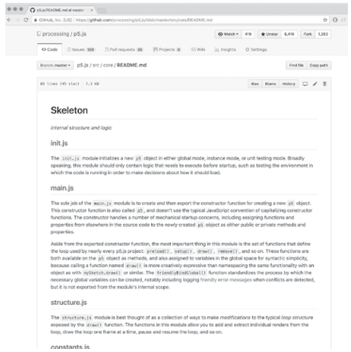
The aim was to make it easier to think through, sketch out, and iterate on ideas. This could range from surfacing available p5.js function parameters through autocomplete to more exotic UI elements overlaid on the code for manipulating and picking values.

A screenshot of the top portion of the p5.js Web Editor. One of the features added is a small "HTTPS" checkbox that switches between HTTP and HTTPS protocols on a per-sketch basis.

FEL 017 A p5.js Dissection Manual, 2018, Vijith Assar, Mentored by Lauren McCarthy.

Vijith Assar treated the p5 code repository as an editorial project, working on internal developer-facing documentation of the architecture and guiding principles in order to help future open-source contributors.

The p5 reference manual is thorough, but it is also primarily aimed at users, which still leaves a barrier to entry for developers interested in contributing to the p5.js core. To help with the latter, Vijith worked on a set



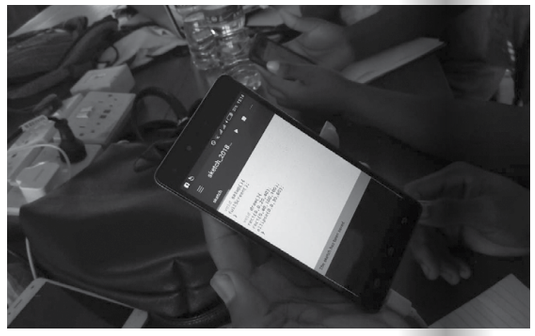

A screenshot of the resulting documentation published on GitHub.

of complementary developer-facing documentation materials covering the architecture and guiding principles of p5.js, spread across a combination of the repository, wiki, code comments, and other sources. Developing clearer explanatory materials made it easier for interested developers to understand the software and subsequently start contributing. This helped speed up the pace of development and invited a more diverse base of contributors.

FEL 018 SuaCode: Breaking the Coding Barrier in Africa with Smartphones,2018, George Boateng, Mentored by Niklas Peters.

George Boateng's project is a pilot of Nsesa Foundation's ambitious program SuaCode, an online course to teach millions across Africa how to code via smartphones. This project aims to take advantage of the proliferation of smartphones to address Africa's digital gap by introducing high school and college students in Ghana to programming using the Processing language.

In this project, George developed a coding curriculum based on the Processing programming language, and then delivered it through a pilot program to 30 high school and college students, who have no programming experience, living in different parts of Ghana. The programming course introduced students to core programming concepts in a visual and fun way through game



Two pairs of hands holding smartphones displaying code for Processing sketches.



development. Curriculum was delivered as an online course primarily through smartphones and also computers via the free learning management system, Google Classroom. Students coded up the assignments using the Processing desktop app (for students using computers) and various mobile-based Processing apps such as APDE (for Android) and Processing iCompiler (for iOS). Students submitted assignments for grading and feedback as they went through the course. The course duration was about 10–14 hours total. At the end of the course, students developed core programming skills and built a game in the process.

FEL 019 Working on the p5.accessibility Project, 2018, Mathura Govindarajan and Luis Morales-Navarro, Mentored by Claire Kearney-Volpe.

p5-accessibility aims to make p5 and its community more accessible to people who are blind and visually impaired through the adaptation, development, and implementation of tutorials, documentation, and learning resources; maintenance of the accessibility features of editor.p5js.org; and the development of a p5 accessibility library.

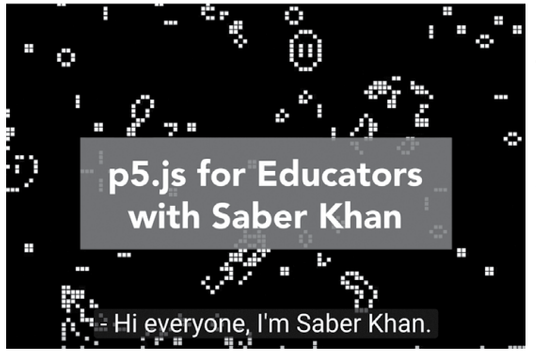
For the past year and a half Mathura Govindarajan and Luis Morales-Navarro have added features to the p5.js web editor that has made it inclusive to people with visual impairments. This resulted in the creation of the libraries, p5-accessibility and colornamer. During the fellowship, the fellows


Screenshot of a tutorial on "Using p5 with a screen reader" on p5js.org.

fully incorporated these libraries to the p5.js widget and p5.js web editor to increase their accessibility. At the same time, they worked on developing accessible learning materials and ensured that the p5.js documentation is fully accessible.

This fellowship continued the Processing Accessibility Project started by Claire Kearney-Volpe during her 2016 Processing Foundation Fellowship.

FEL 020 Building Inclusive Learning Spaces for Students and Teachers, 2018, Saber Khan, Mentored by Daniel Shiffman.




A screenshot of a YouTube video with a black background with white pixel-y shapes. A hot pink box with white type reads, "p5.js for Educators with Saber Khan." The caption at the bottom of the video reads, "Hi everyone, I'm Saber Khan."

Saber Khan's fellowship project was to develop an Education Outreach Manager Role at Processing Foundation. The goal of the role was to support the Processing Foundation's engagement with K-12 educators. Saber created an online community for teachers, published curriculum on the Processing Foundation website, managed relationships with K-12 educators, and brought CC Fest to more students and teachers.

With the growth of CSforAll, there is a large demand for tools and curricula that students and teachers can use. Currently, most of the available curricula are offered either from for-profit ed-tech startups or nonprofits backed by the technology sector. This has led to the development of a CSforAll movement that has inherited the biases of the industry, while it tries to solve challenges around equity, access, and power. The Processing Foundation and the community of educators around it are well poised to help build an independent K-12 CS community and movement that takes seriously questions of ethics, identity, and creativity. The community has a large group of committed educators, a commitment

to diversity and inclusion, and a critical perspective on biases and its impact on the marginalized. An Education Outreach Manager can help coordinate this effort to build better CS education for K-12 students and teachers.

Saber began by sharing existing p5.js and Processing curriculum on the Processing Foundation website, creating an online community for educators on Twitter and elsewhere, organizing in-person events like CC Fests, and managing relationships with K-12 educators.

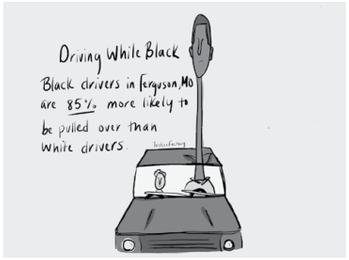
FEL 021 p5.js 的中文翻译 — 为支持更多种语言翻译做准备 (Chinese Translation for p5.js), 2018, Kenneth Lim, Mentored by Xin-Xin.

Kenneth Lim's fellowship project aimed to provide a translation of the p5.js website and documentation into Chinese, breaking down the language barrier to learning to code, and providing a starting point to maintain said translation.

Rather than trying to include people while expecting them to know English, an inclusive and diverse community should speak multiple languages. It's a bit of a stretch to expect people to understand many languages but it is entirely possible to have softwares speak multiple languages, once the effort is put in to make it so. The hope was that this project will eventually lead to many more translation efforts.


English and Chinese text on the homepage of the p5.js website.

FEL 022 Justice Factory: Data Visualizations For Empathy, 2018, Ari Melenciano, Mentored by Daniel Shiffman.




An illustration of a white male and Black male driving a car. The metric unit is the length of each driver's neck, representing the disparity between races when being pulled over by police. The Black male's neck is figuratively 85 percent longer than the white male's neck.

Justice Factory is an interactive data visualization tool that highlights social justice issues and human rights violations, with the intention to serve as a tool to advance the fights of activists.

Ari Melenciano has always believed that data visualizations are a language of the oppressed: she recognizes the power of data, particularly the transformation of data being turned into easily digestible forms of information and its potential to reflect the truth. Through interaction with data visualizations, the arguments of activists are advanced, the awareness of the general public is increased, and the oppressed are able to more easily identify that the injustices they are facing have a name, commonly experienced by others, and are part of a system, not just coincidental. Justice Factory is a library of design-centric, engaging, and interactive data visualization templates built with Processing.

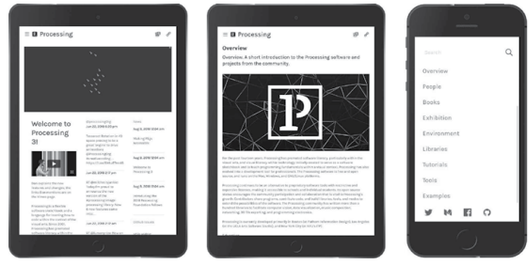
FEL 023 Teaching Processing in AP Computer Science Courses, 2018, Kaitlyn M. O'Bryan, Kate Lockwood, and Thomas J. Reinartz, Jr., Mentored by Casey Reas.

The APCS-A course covers a subset of the Java programming language, which can be difficult for students to engage with. Processing can support a wider range of

Four students at a desk, working on computers.

students in understanding the content, allowing teachers to address the same course objectives. This project created four chapters of an open-source, project-based textbook in Processing that is aligned with the APCS-A outcomes. The textbook was released






Screenshots of the Processing.org website on tablets and iPhones.

on GitHub with the end goal to create a community supported APCS-A curriculum in Processing that is perhaps more accessible and engaging to a wider variety of students.

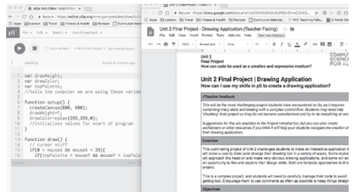
Each chapter centered around an open-ended project in Processing that aligns with APCS-A objectives. Along with the projects, they provided high-level lesson plans that provide: outcomes for the lesson, alignment with APCS-A objectives, project rubrics, and links to additional resources.

FEL 024 Renovating the Processing.org Website, 2018, Kirit Tanna, Mentored by Scott Murray.

Kirit Tanna's fellowship project was to upgrade and enhance the overall user-experience on the processing.org website and its offshoots to be consistent, responsive, accessible, and forward-looking. This enabled users to explore the wealth of information on these websites. In addition, the website accepted contributions from the community to support its growth as a platform and served as a source of inspiration to its users.

Planned upgrades affected: blog, news, and social integration; reference and examples; support forums; exhibition; publications; and the shop. Design elements focused on responsive design so as to access the site via mobile devices, thus improving overall accessibility; improving content visibility, social/community features, and overall loading time; and developing a style guide and listing UI components required across the site.

FEL 025 Making Processing Available in NYC Schools, 2018, Courtney Morgan and Jose Orea, Mentored by Saber Khan.



Screenshot of two browser windows. On the left is a browser open to the p5.js Web Editor, with code for a drawing application; on the right is a browser open to a Google Doc showing a lesson plan for coding a drawing application in p5.

José Orea and Courtney Morgan worked with the NYC Dept. of Education to make their Introduction to Computational Media high school curriculum better to educators everywhere. It was shared publicly that fall on p5js.org.

FEL 026 p5.js 1.0, 2019, Stalgia Grigg and Evelyn Masso, Mentored by Lauren McCarthy.

Evelyn Masso and Stalgia Grigg worked alongside the larger community of open-source contributors to realize a 1.0 release of p5.js.

Stalgia Grigg, Evelyn Masso, and Lauren McCarthy work with five others on laptops at desks and tables in a white room.

Stalgia kicked off the fellowship with a focus on WebGL stability improvements, expanding developer documentation, and creating new channels of communication for contributors.

Evelyn also focused on various tooling in the p5.js repository to improve the p5.js contributor experience.

FEL 027 Expanding SuaCode to Teach African Students to Code on Smartphones, 2019, Prince Steven Annor, Mentored by George Boateng.

During 2018, Processing Foundation fellow George Boateng piloted SuaCode, a smartphone-based coding course with 30 high school and college students in Ghana. Having assisted George on the project as a member of his team, Prince Steven Annor's project sought to build upon that work and scale it to more people. In this project, I developed an automatic grading system, recruited mentors, and delivered a smartphone-based online coding course to introduce programming using Processing to 100 high school and college

students in different parts of the African continent. Students can access the course lesson notes on Google Classroom, a free learning management system, and then program their assignments using the Android Processing Development Environment (APDE) app. The programming course introduced students to fundamental programming concepts in a visual and fun way through the development of a Pong game. A GitHub repository that contained sample Processing sketches which illustrated concepts in the course were provided to students for cloning their APDE app. Students submitted assignments for grading and received feedback as they went through the course. At the end of the course, students developed fundamental programming skills and built a Pong game using Processing.

A student doing his assignment on his phone.

FEL 028 Making Processing Tools Accessible to the Indian Community, 2019, Manaswini Das, Nancy Chauhan, and Shaharyar Shamshi, Mentored by Mathura Govindarajan.

Based on the most recent UN data, the population of India is estimated at 1.35 billion and 80 percent reads or understands Hindi.


A teacher points to a projection of the p5.js website at the front of a classroom. Rows of students with open laptops look on.

Approximately more than 85 percent of India's population lives in the lower or middle income bracket, which often doesn't have the resources to afford classes for computer programming. We used Processing to serve the Indian community, by translating the p5.js website and documentation to Hindi, providing p5.js YouTube tutorials in Hindi, and reaching out to non-governmental organizations that emphasize and impart software literacy among underprivileged students.

FEL 029 An Immediate Graphic User Interface Library for Processing, 2019, Doeke Wartena, Mentored by Casey Reas.

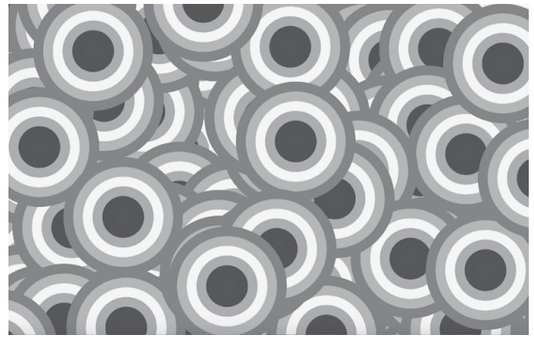
Doeke Wartena worked on a library for creating a Graphical User Interface (GUI). Instead of the traditionally retained mode GUI system, this library is using an immediate mode (IMGUI) where the whole UI is created each frame. The IMGUI was an idea of Casey Muratori back in 2002 and was slowly getting more momentum. This immediate mode avoids a per UI element initialization and the need of registering callbacks. It allows for elements to be created on the fly, which allows for more dynamic interfaces. To give a concrete example, "if (on(CLICK, button())) println("!")" would create a button and when clicking it would print "!" in the console. My goals were to make people more productive by implementing graphical user interfaces instead of making them less productive by implementing graphical user interfaces. Along with this, making custom styles should be easy and understandable without the need of reading a lot of documentation. I was hoping to address issues where the web world took a wrong turn.

A graphic in gray, white, and blue, that shows the capital letter P drawn inside a blue square, inside a gray rectangle. The image's pixels are made up of a graphic of a rectangular ellipse, with a circle inside of it on either the left or right end.



FEL 030 Pride Peers Programming Prosperous Protection, aka P5, 2019, Matilda Wysocki, Mentored by Daniel Shiffman.

Matilda Wysocki spent time teaching trans and gender nonconforming youth basic programming and design in a queer, privacy and digital literacy context. There was support and resources for those who wish to deepen their skills as developers afterward, to have a tool for self expression and community building, or to just understand the forces shaping our world a little better. This project consisted of six-hour classes on weekends, but also depended on the schedule of the space and students. The space was the Ali Forney Center. To supplement the programming and art experience, Matilda included guest speakers to discuss AI and its impact on society, personal digital security, online harassment, mass surveillance, automation, and work. The class projects were adapted to fit the class's interests and to help facilitate discussions around the above ideas. Matilda was originally asked to teach programming as part of a budding multimedia training initiative for TGNC youth, and their hope was that this was the first of many classes as a part of such a program.



Many circles drawn within each other overlap throughout the image. Their rings follow a color pattern, with purple at the center, then blue, green, yellow, orange, and red.

FEL 031 Expand and Diversify p5.js in China, 2019, Qianqian Ye, Mentored by Dorothy Santos.

My project aimed to make p5.js more accessible in China, especially within underrepresented women's and non-male identified groups. After having the p5.js website and documentation translated into Chinese in 2018 by Foundation Fellow Kenneth Lim, we had more work to do to activate and cultivate the young Processing and p5.js communities in China. To counter the fact that most online educational resources such as YouTube are banned in China, I recorded p5.js video tutorials for beginners in Chinese and shared them on Chinese video sites. I also partnered with other female Chinese creative coders to host p5.js workshops for girls, women, and other non-male identified people in China, as well as posted interviews with role models in the Processing and p5.js community on Chinese social media. Throughout the process, I explored socially conscious, culturally sensitive, and non-Western models of teaching creative coding. By teaching women and non-male identified people p5.js, we promoted diversity and activated marginalized communities within China in new ways. I hope this project inspired people from other minority groups in China to participate in the creative coding community.


A 4x4 grid of screenshots of Qianqian's p5.js tutorials. The top-left is a logo that reads "Qtv" in a purple circle. The top-right caption reads: "1 Minute Videos in Mandarin / Creative coding, art, technology / Available on Bilibili, TikTok, YouTube, and Instagram."

FEL 032 Coding with Sound and Art for Middle-School Students, 2019, Layla Quinones, Mentored by Saber Khan.

A teacher points to a projection at the front of a classroom. Three individuals with open laptops look on.

This project consisted of writing a curriculum that taught students how to integrate sound, animation, and interactivity into a creative computational artifact in p5.js. Layla Quinones focused on developing tools for teachers who have little experience teaching topics in CS to urban communities.

FEL 033 Teaching Digital Dance: Coding, Graphic Design, Animation, Dance, and Robotics , 2019, Emily Fields, Mentored by Saber Khan.

An individual holding a closed umbrella looking up at a projection of raindrops.

Emily Fields and her students experimented with motion detection. They created an interactive projection that dancers used in their school's annual Digital Dance performance. Emily used her findings to create a unit of study on motion for the NYC DOE to share with fellow computer science teachers.

FEL 034 Make-IT V2: Enabling universal access to technology, 2020, Abdellah Iraamane, Mentored by George Boateng.

Two children, one of whom is wearing red, draw on white paper using paint and colored crayons.

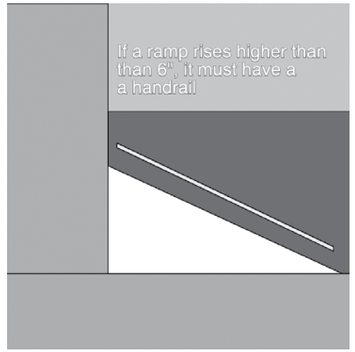
We explored the advantages and challenges of learning to code for first-timers using creative coding techniques vs. the traditional "tutorial-based" model. We hypothesized that using creative techniques to teach code will result in fewer dropout rates, more engagement from communities who have not learned to code before, and more motivation for educators to be creative in their teaching ways, especially in cases where they themselves will have to learn to code before teaching children how to code.

This project focused on a specific community landscape with very little to no knowledge of code, in remote villages in the rural southern regions of Morocco. The context there was particular: There were no educators qualified to teach code in the traditional model. The context of the project was also unique in the sense that the target community does not speak the official language of the country, and hence we researched the possibility of teaching both the language and code through Processing software simultaneously.

In the grand scheme of things, we wanted to achieve universal, long-lasting access to code, in contrast to the traditional solutions that eventually fade because the learning curve becomes too steep to follow.

FEL 035 Know Your Rights, 2020, Kalila Shapiro, Mentored by Luis Morales-Navarro and Claire Kearney-Volpe.

I used Processing (specifically the p5.js library) to develop a website called knowyourrights.com, which helps people with disabilities understand their rights in clear and simple visualizations. I highlighted the disparities in employment and quality of life between the disabled and abled communities, and made people feel comfortable standing up for themselves when the laws that protect them are being violated. Despite the fact that the Americans with Disabilities Act (the ADA) was signed into law nearly 30 years ago, only 56 percent of disabled Americans are employed.




A static image. Set on a blue background is a gray building sitting on a field of grass. Next to the building is a red wall. Against the building is a white ramp and slightly above it is a thin yellow handrail. There is text on the screen reading "If a ramp rises higher that 6," it must have a handrail."

knowyourrights.com is an interactive online space that clearly and concisely displays different aspects of the law in both text and visualizations – with specific focus on the sections of the ADA that protect students and Section 504 of the Rehabilitation Act of 1973. This section was the main focus because education remains largely inaccessible for many students with disabilities. This is a law visualization project, and is a helpful alternative to sites like a data.org and ada.gov, which are dense with legal terms. It is also web accessible according to W3 standards.

FEL 036 Cozy Coding, 2020, Aren Davey, Mentored by Daniel Shiffman.

Cozy Coding is a series of cozy weekly Twitch streams that hosts interactive p5.js tutorials and lessons. For each stream, there is a chosen topic which Aren Davey researches and demonstrates to viewers. Viewers can code along with me via the p5.js editor, and ask Aren questions in the chat and receive answers in real time. The environment of the streams is relaxed, warm, and nonchalant – this isn't a boring class lecture – so there is a lot of freedom for the direction of the stream. Some of the chosen streams are collaborative live-coding jam sessions, where the viewers are able to code with me on a single project and create something interesting together.

Many hexagons of varying shades of yellow and green are arranged over a grey background. The code that generates the image is overlaid on top in white and green text.

FEL 037 Open Computer Vision for p5.js and Processing, 2020, George Profenza, Mentored by Golan Levin.

The latest version of OpenCV has a lot of extra functionalities. One important feature is the DNN module, which makes it easy to interface with deep neural network frozen graphs from major frameworks such as TensorFlow and Caffe, accessing networks like YOLO, OpenPose, Face Landmarks, etc.

For interactive art installations and other creative projects, George Profenza believes access to an updated free open-source library is beneficial for the Processing community. This is for Processing Java mode or p5.js using OpenCV.js.


Sparse optical flow example displaying green points overlaid on top of the porcelain cat and book. The green points represent tracked features.

Such an update is a good foundation for teaching workshops, covering areas such as:

- image filters (e.g., understanding convolution/making custom kernels)
- computer vision basics (frame differencing/bg subtraction, color tracking, blob detection, etc.)
- advanced topics (image classification, facial landmark detection, pose estimation, etc.)

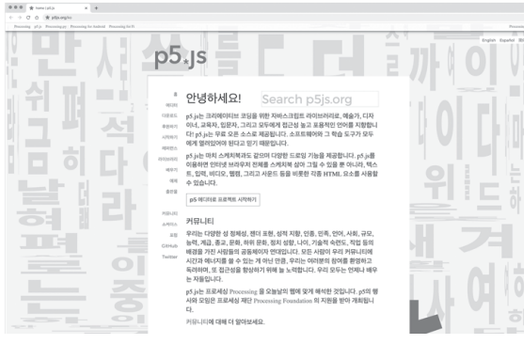
FEL 038 Creative Coding & Computational Thinking for Young Students, 2020, Michael O'Connell, Mentored by Layla Quinones.

This project sought to curate, create, and pilot p5.js activities and lessons to help children ages 8–12 more easily learn coding and computational thinking. Michael O'Connell hoped to better engage students by inspiring them to create visually, and to develop script-based coding skills, and thus go beyond Scratch and code.org. Michael drew from their training and experience with p5 and years of teaching coding to grade-school students, starting with activities like the "first sketch" exercise in the p5js.org Get Started page. Michael referenced the work of other Processing Foundation Fellows, including Niklas Peters and Layla Quinones, and incorporated simple.js and other libraries to help make p5.js easier for new young coders before they reach middle school. Michael then employed these exercises with the coding clubs they led, allowing them to pilot, iterate, and collect feedback from 100+ students. The goal was to make coding and computational thinking more broadly accessible and to better reach younger students, for whom a more traditional script-programming curriculum may not resonate, including those who are eager to learn and have unbounded ambition and seek a

Zoom tiles of 17 individuals.

next step beyond the more structured code.org activities and the block-based Scratch programming environment.

FEL 039 p5 for 50+ (p5 for 50 and Beyond), 2020, Inhwa Yeom and Seonghyeon Kim, Mentored by Qianqian Ye.




A screenshot of p5js.org/ko, the Korean translation of p5js.org, with Seonghyon Kim's "Hunminjeongeum 2020" as the background artwork.

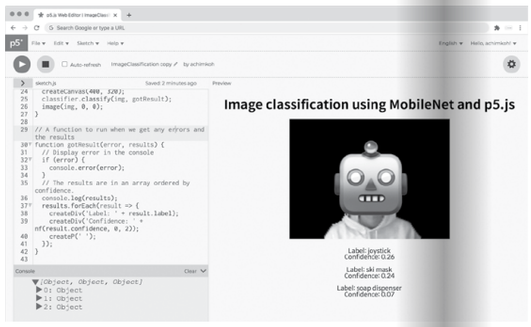
The project p5 for 50+ (50 and beyond) aimed to improve the digital literacy and rights of middle-aged and older adult creators in a non-Anglophone, aging, yet highly digitized and tech-centered society: South Korea. Along with the Korean translation of the p5 website, Inhwa Yeom and Seonghyeon Kim hosted a weekly workshop that specifically targets participants of the upper age group who face even greater linguistic and emotional barriers than younger generations when it comes to learning coding.

Experimenting with and designing learning methodology for people around and over 50 is the most important challenge to this project. Throughout the Fellowship, Inhwa and Seonghyeon developed and evaluated various learning interfaces and interactions, logistics, and environment settings of their workshop, with aural, audio-visual, contextual, and interactive elements of p5. As one methodology, p5 and creative coding were contextualized at the workshop as a medium for addressing personal and social issues constructed from the participants' technological surroundings. With the p5 artworks they create, the participants were expected to imagine their own online gallery, agora, asylum, archive, and marketplace where they could voice their experiences of being 50+.

FEL 040 Critical Machine Learning with ml5.js, 2020, Achim Koh, Mentored by Joey Lee.

Critical Machine Learning with ml5 is a series of educational material that enables a critical understanding and use of machine learning and artificial intelligence, with a focus on the social implications of said technologies. The material is produced in Korean and English, for local usage and broader access.

The developed material is activated through a series of workshops in Seoul. The workshops are accompanied by a discussion-oriented event around critical perspectives on software, ML, and AI. All events aim at fostering a more inclusive environment in the Korean tech context.




A web browser displays the p5 web editor. The source code is displayed on the left side of the screen. The result of the executed code is displayed on the right side: the title reads "Image classification using MobileNet and p5.js." An image of a person from the chest up, whose face is covered by a robot emoji, is shown. Underneath, the caption reads: "Label: joystick, Confidence: 0.26, Label: ski mask, Confidence: 0.24, Label: soap dispenser, Confidence: 0.07."

The project is partly a response to industry- and government-driven AI initiatives, which typically cast AI-related education as vocational, as previous coding education initiatives tended to do with software programming. Ultimately, this work builds resources that help people think critically about technology, power, and knowledge.

FEL 041 DIY AI: ml5 Community Starter Kit, 2020, Emily Martinez, Mentored by Lydia Jessup.

DIY AI: ml5 Community Starter Kit is a toolkit that teaches beginners how to set up and train an artificial intelligence using ml5. The kit includes instructions for how to find, clean, and process data to feed into a machine-learning algorithm. The step-by-step guide also includes guidelines for running small workshops where people can teach each other how to design and build their own artificial intelligence without having to rely on "experts." The first training module is focused on text-based, conversational chatbots using multi-layer Recurrent Neural Networks (LSTM, RNN).

As co-creator of Queer AI and a member of Color Coded, Emily Martinez is motivated to develop tools and materials for queer, BIPOC, and other marginalized communities interested in working with technologies like artificial intelligence for their own purposes, pleasure, and empowerment. As critical conversations continue to unfold around the limited access, risks, and harms of artificial intelligence, their hope is that together, we can build counter-examples tending to the poetics, play, self-discovery, and world-building potential of artificial intelligence with curiosity and care.


Webpage reads DIY AI: ML5 Community Starter Kit.

FEL 042 Fine-tuning ml5.js: Friendlier Code & Development Processes, 2020, Bomani Oseni McClendon, Mentored by Joey Lee.

Bomani Oseni McClendon worked with mentor Joey Lee to contribute a variety of improvements to the ml5.js core library and website. As ml5.js matures, they were focused on stability and addressing community needs, with the goal of making ml5.js more accessible to a wider variety of contributors and creators.

Our changes began with a simplification of the release process for the library so that contributors can more easily release new versions of ml5.js. We focused on improving the parity between our library features and our documentation, and additionally, made improvements to our automated testing workflow. Lastly, we built tools to improve our integrations with the p5.js Web Editor, Teachable Machine, Typescript-based projects, Node.js, and more.

An additional focus for our work related to increasing the visibility of our values and community statement, which included a variety of improvements to the ml5.js website. Throughout the project, Bomani helped to support our community's requests through GitHub issues and push improvements to code quality and design pattern consistency throughout the library. We believe that these changes would enhance the experience for people using ml5.js and open source contributors.

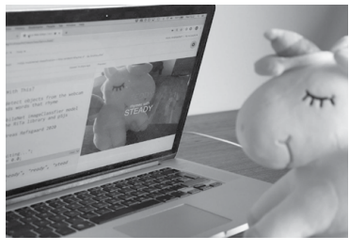
Automatic facial occlusion using the ml5.js Facemesh model and p5.js. Original base photo by Edrick Chu.

A person sits in a chair. Their entire face is covered by a black shape, with white dots positioned over key facial features.

FEL 043 ml5.js Examples,2020, Andreas Refsgaard, Mentored by Yining Shi.

I proposed to make a new set of playful interactive examples to support the ml5.js library, website, and community. The aim was to showcase the creative potential of using ml5 and attract even more people to use the library through a series of simple and more advanced examples:

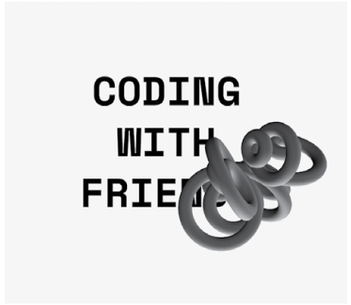
- Simple interactive examples – Some people might be interested in a particular technique, but do not know how to get started with it. Other people do not fully understand certain techniques or know what to build with them, and need to see them in action in order to come up with ideas of their own. These simple examples build on top of existing ml5.js examples, by adding just enough additional code to make them more playful or engaging.
- Advanced artistic examples – The more advanced examples go a step deeper, either by combining multiple functionalities within ml5.js or by using p5.js to make the outputs from ml5 machine learning models even more interesting. Compared to the simple examples, these examples generally are longer and more complex, and



A laptop shows ml5.js code on the left of the screen, and an image output. A small keyboard is shown to the left.

therefore are intended to serve as inspirational examples rather than as tutorial examples for beginners.

FEL 044 Coding with Friends, 2021, Ambika (Computational Mama), Mentored by Shaharyar Shamshi.



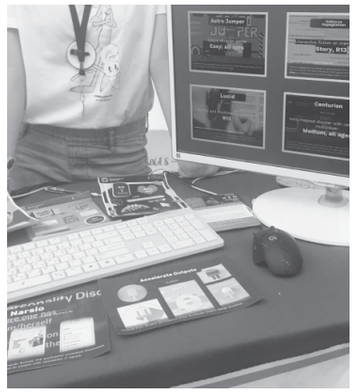

Square image with ivory background. “Coding with Friends” is written in a large size in the center with some torus and ring shapes floating around the words.

What does it mean to have a simple intimate chat over a shared activity? Does creating with code only serve its purpose towards an output? Can the act of two friends coding together mean more than a product, a production, or a performance?

Coding with Friends is an ongoing series of livestreams with womxn makers and creators. These are one-on-one sessions, where Computational Mama invites a friend to code together. A casual act of coding together can be equated to making dim sums or folding paper cranes or even playing a board game.

Coding with Friends casually and simply claims space for womxn creators. The series extends the idea of co-coding as a form of camaraderie, friendship, and self-care. Coding with Friends reinforces ideas of care, sharing, and connectedness. What could be more exciting than extending warmth and care through a positive shared experience? What could be more provocative than to build soft, understated, and rooted networks of womxn creators? What can be more powerful than relationships built in safe, creative environments?

The project gives an opportunity for womxn creator communities and practices around creative computation to move away from notional ideas of what code can produce, who can produce code, and where it can be produced.



Developh’s booth at the 2018 Manila Maker Faire. A pink keyboard and monitor are set up on a bright blue table, with piles of sticker sheets and printouts with project details on it. A color-blocked sticker set of programming jokes (one is a heart with “The Art of Computer Programming”), a light pink and blue set with “Girls Byte Back,” and a dark blue set with a “RIP semicolon” on a gravestone. On the monitor are demos by Developh members: “Astro Jumper Space Shooter Game,” “Halina sa Hapagkain.”

FEL 045 Developh p5.js Camp, 2021, Chia Amisola, Mentored by Dorothy Santos.

Over 50 percent of information technology graduates in the Philippines are reported to lack programming competencies, and there is an absence of expository, pre-college technology and art programs in the Philippines. Developh Camp builds on the work of a community not-for-profit that has been running education and innovation programs to boost the Philippine technology ecosystem since 2016. Developh Camp cultivates a lifelong community for mission-driven technologists and creators while producing collaborative projects and resources for the larger nation. With focus on peer and small slope mentorship, emphasis on teaching non-technologists code, and initiatives rooted in social impact, Developh Camp uplifts both the art and creative code scenes in the Philippines.

Developh Camp supports a cohort of high school/college-aged Filipino technologists in producing socially driven, artful technology projects built with p5.js. At the end of the Camp, an online exhibit was launched to host their work. Outputs included biweekly workshops and teaching materials open to all (in both English and Filipino), co-creating streams and events, and the framework for running replicable community events sensitive to low data spaces.



After Camp, fellows continued to be integrated into the Developh community as they further explored technology, design, and founderhood. p5.js is integrated into our curriculum, discussion clubs, projects, and more.

FEL 046 PORTAL.WEB – A Cyber Witch Coven, 2021, Cy X, Mentored by Johanna Hedva.


Cy, a dark skin agender person, is sitting on the floor wearing a pierced light grey halter top and bottoms while holding a lightly lit piece of paper of something they’d like to release. In front of them is a candelabra with five white lit candles, crystals, and an abalone shell. The ritual video is framed by the image of a dessert and words on the bottom of the screen that reads “Release the myth of independence that restrain me from seeking help.”

PORTAL.WEB is a cyber witch coven that reimagines our cyber powers through the use of emerging, indigenous, and ancestral technologies. The coven seeks not to establish a hierarchy of technologies but rather to work with cyber witch practices in a way that intimately engages our own radical imagination, ancestral knowledge, and inner-knowing.

Cy X researched cyber witch practices and theories in order to develop a year-long syllabus, cyber witch workbook, and series of workshops, performances, collaborative rituals, and ceremonies, which will invite the collective to not only interrogate what is / is not technology but also to explore new ways of learning, breaking, and hacking to create tools of our dreams, ones that could open new portals and celebrate Black, brown, queer, and trans folks.

A lot of technological education currently seems to negate knowledge from other frameworks, cultures, and ways of thinking. This project, through acknowledging “a world of many worlds,” sought to work within the framework of the pluri-verse, while also acknowledging that those at the margins have often been negatively impacted by technology. This project acknowledges that any form of technology education needs to center healing at its core, by connecting with our inner selves, including care and love at the center, and uplifting ancestral knowledge, while imagining and creating new worlds.

FEL 047 Pê Cinco: Internationalization and Popularization for Portuguese Speakers, 2021, Felipe Santos Gomes, Julia Brasil, Katherine Finn Zander, and Marcela Mancino, Mentored by Claudio Esperança.

Brazilian coders, artists, and researchers who have self-taught to code, have experienced firsthand the difficulty in finding educational content in our native language, Portuguese. This project came from the desire to make p5.js accessible to the Portuguese community by translating the p5.js reference. Along with the translation, they produced a series of short videos presenting the project and its main features.

Like previous Processing Foundation Fellowship projects, translation can open up ways for more people to contribute, and in doing so expand the community, reaching new audiences that otherwise would not be able to access the tool in its full potential. This effort aims to encourage other projects, translations, and educational projects in Portuguese.


The p5.js home web page with a pink circle surrounding the logo and an arrow pointing to the Portuguese pronunciation of “p5” into “pê cinco.” Next to the word “Hello” is a translation to Portuguese “Olá” followed by a smiley face.

FEL 048 Indigemoji, 2021, Akeltye-antheme awetyeke, Mentored by Yining Shi.

Akeltye-antheme awetyeke – “teaching to listen” – explored what it would take to teach machines to listen and understand Arrernte, a traditional Indigenous language of

Central Australia. By creating a small sovereign dataset of Arrernte spoken words, we created a web-based prototype in p5.js and ml5.js to encourage young people to speak their own language. In doing so, we hope to start a conversation with machines, to translate and explore concepts of machine listening, and to consider what the implications and potential of that might be.

Emoji bosses Kathleen Kemarre Wallace, Veronica Perrule Dobson, and Joel Liddle Perrule. Three people sit at a table, laughing.

FEL 049 Internationalization Support: Spanish Localization for the Processing website, 2021,Omar Verduga, Mentored by Esteban Sandoval.


The Processing Foundation logo, the letter P, is accompanied in the top-left corner with the word Hola, Spanish for Hello. In the bottom-right corner, there is the letter ñ, pronounced "Enyay," which has come to represent the identity of the Spanish language.

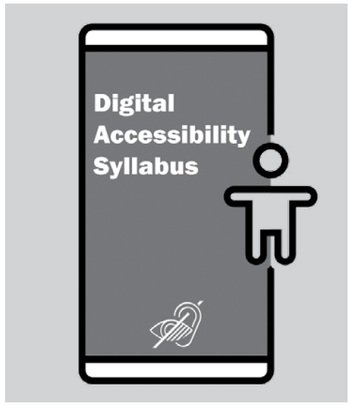
Currently, the p5.js and p5.js web editor websites support translations for different languages; however, the Processing website includes no translations (there is a pull request with a German version, which is indicative that some other users are aware of the opportunity). During the GSoC 2020 program under the Processing Foundation, Omar Verduga completed an internationalization solution for the p5 web editor, where internationalization was studied from a broader context, not only on the technical side. Inclusive and non-gendered language, and accessibility through ARIA labels were topics discussed with the community, and a localized version of the p5.js web editor was delivered in Spanish. The goal of this Fellowship project is to join the current efforts in translating the website and create guidelines for future contributing members in how to organize the translations, while delivering a Spanish localized version of the Processing website, with the idea of making it more accessible to the international community and to further promote its usage among students of Spanish (Latin American Spanish, particularly) speaking communities.

FEL 050 Digital Accessibility Syllabus, 2021, Adekemi Sijuwade-Ukadike, Mentored by Claire Kearney-Volpe.

From Adekemi Sijuawade-Ukadike's observation, there are initiatives on a university level to bring awareness to digital accessibility. Kemi wanted to add to this by creating a university-level, open-source course model that directly speaks to why it is important, as well as how to create accessible digital content from the inception of a project, rather than as an after-thought once the project is close to completion.

The COVID-19 pandemic has shifted most of our communication into the digital realm. Now more than ever, about 61 million Americans who live with a disability are relying on digital spaces as an important means of connection and survival to health, financial, social, and job-related services and information.

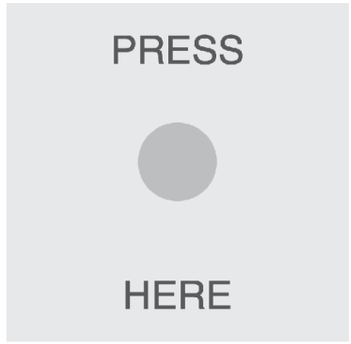
Kemi believes that if a full-semester immersion to digital accessibility is an option for those studying interactive and immersive experiences, an awareness that the DOM, as well as libraries such as p5.js's accessibility are tools and how to use them – would be super useful to interaction designers, web developers, coders and those creating digital experiences. Perhaps we can see an influx of robust digital projects, not only to avoid litigation, but for better interactions overall.




A graphic of a handheld device with a green screen and the words "Digital Accessibility Syllabus" in white font. Accessibility icons in black and white.



FEL 051 Starter Kit for Teaching Creative Coding with p5.js, 2021, Angi Chau, Mentored by Saber Khan.




The text "Press Here" is displayed in dark blue and a yellow circle in the middle of the screen. The word "Press" is above the yellow circle and the word "Here" is below the yellow circle.

As more CS classes are being implemented across the world (yay!), Angi Chau noticed that there are many teachers who are tasked with teaching CS, but who never received formal training in CS. While there are pre-made curricula that educators can use (e.g., Google CS First, Code.org), Angi wanted to investigate the space between these "plug and play" options and a total blank canvas, and explore how to offer easily adaptable modules for new teachers to build their own creative coding curriculum.

Angi wanted to create a "starter kit" to bootstrap teachers new to CS, which will likely be some combo of online resources, videos, lesson ideas, and "poetic prompts." She imagined three main areas of this kit:

Learn – curated resources for teachers to learn/practice a CS concept themselves,

Teach – compilation of new and existing resources (videos, slides, lesson plans, activities) offering different ways to teach a concept in the classroom,

Create – open-ended prompts for students to practice this concept while still allowing for their creative voices to shine through.

FEL 052 Cultivating Creative Connection with Scratch and p5.js, 2021, Shawn Patrick Higgins, Mentored by Saber Khan.


An animated cartoon GIF of Shawn Higgins clicking his heels together and whistling plays in the center of his scratch bio page/website. He wears a red cap and has a large beard. He wears blue jeans, an orange sweater, and red shoes. There are four buttons on the page: About, Fun Facts, Likes, and FAQ. A large yellow "3-2-1 hi everybody!" animates above the small cartoon. The entire animation takes place within a grey scratch window.

This project was all about harnessing Shawn Patrick Higgins' students' all-time favorite Scratch experiences and creating a curated project-based pathway for students to experience the best of the best: first in Scratch, then by "upmixing" them into p5.js. Through video tutorials paralleling the recreation of classic Scratch projects like Natalie Rusk's "Paint with Gobo" and Ipzy's "Bio Page," as well as Shawn's own popular motion graphics and sound board projects, students were introduced to Scratch and p5.js through step-by-step visual instruction, with a focus on the familiar and functional reapplication of concepts.

Shawn referenced the work of other Processing Foundation Teaching Fellows – including Michael O'Connell and Layla Quinones – on how to best enhance the onboarding process for middle school students.

By designing explicit creative parallels, Shawn's goal was to inspire both Scratch students of moderate proficiency to take the jump into p5.js, as well as appeal to inexperienced coders who may be intimidated by syntax, to begin their journey into digital making with a visual language like Scratch!

FEL 053 P5LIVE – Walkthrough, 2021, Ted Davis, Mentored by Saber Khan.

For two years I've been developing P5LIVE, a p5.js coding environment designed for VJ, live-coding, remote collaboration, and teaching. Initially designed for fullscreen algorave usage, COCODING (shared document editing) was added early on, opening the door for both local and remote jams. Initially a fun feature, it's become a crucial tool for teaching introductions to creative coding in these COVID-19 distant-learning times. Throughout the pandemic, it supported peer


Screenshot demonstrating concept of COCODING within the P5LIVE environment. Code overlays 3D rendering of p5.js sketch, interface allows sharing options.

collaborations, entire classroom gatherings, coded A/V performances, and audio-reactive remote meditation sessions. Extra features geared towards learning and performance include Auto-Complete of p5.js functions, Smooth-Compile for additive changes to the sketch output, and SyncData enabling sharing of local data (MIDI, sensors, etc). Over the next year, additional features to support remotely teaching code, P5LIVE Classroom, will be developed through an FHNW Lehrfonds grant.

While P5LIVE's features are documented in a README of more than 2,000 words, they have yet to be captured as video walkthroughs with example usage. Through this Fellowship, all current and upcoming Classroom features were demonstrated and shared freely as a video tutorial series. Aimed for teachers, students, and performance artists, ideally exposure to this tool and features helped them throughout and beyond remote creative coding times.

Learning to code is difficult and it takes time. Processing is dedicated to making learning to code more accessible. We feel everyone should have the opportunity to learn and have access to resources. Our software and learning materials have always been free to use and also free to explore and modify. We aim for software literacy, meaning teaching people how to read and write software. Processing is unique in its approach to learning how to code within the visual arts and in our aim to teach coders more about the visual arts. Processing has reached not only artists and coders, but also designers, filmmakers, educators, musicians, performers, and students of all kinds.

EDU 001 Eclipse Replacement For Processing Development Environment (PDE) with Localization, 2011, Harshani Nawarathna.

The project idea suggested a new Processing Development Environment (PDE) based on Eclipse and Xtext to handle syntaxes.

EDU 002 Dynamic Library and Tool Tool Loading, 2011, Peter Kalauskas.

The goal of this project was to ease the process of installing and updating contributions by adding a library and tool manager to the PDE.

EDU 003 Debugging Tool for the Processing Development Environment, 2012, Martin Leopold Groedl.

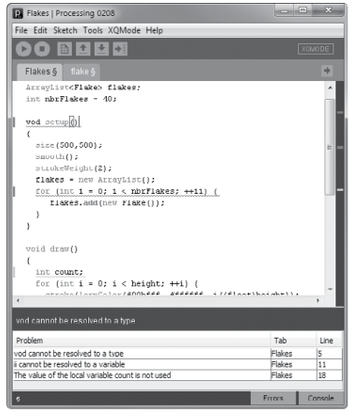

A hand-drawn bug is centered in a circle enclosed in a larger square. Each element in the logo is a different shade of grey. There is text directly below the square that reads “Debugging Mode for Processing.”

The goal of the project was to develop a working prototype of a tool for debugging Processing sketches.

EDU 004 Sketch Assistant, 2012, Manindra Moharana, Mentored by Daniel Shiffman.

This project brings real-time checking and reporting of errors and warnings along with additional features to the PDE.

EDU 005 Streamlining Bluetooth Connectivity in Processing for Android, 2012, Joshua Albers.




The Processing text editor with set-up code. Some of the code is underlined in red to exemplify the real-time checking feature. Errors and warnings found in the code are also displayed in a console at the bottom of the text editor.

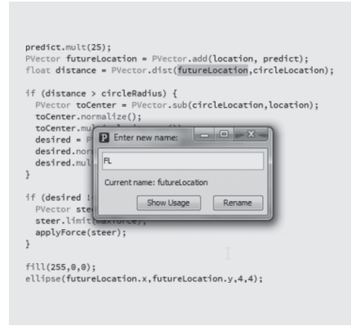
I proposed developing a library that streamlines basic serial communication over Bluetooth for Android so it’s as easy as the desktop Processing application or on an Arduino device.

EDU 006 GUI Library for Processing, 2013, Martin Grödl, Mentored by Florian Jennet.

The goal of this project was to create a functional prototype for a new Graphical User Interface Library for Processing.


A 2x2 grid displays four different themes of the proposed GUI. Each theme has the same UI elements (sliders, toggles, buttons, labels) but varies in color. The top-left theme has grey hues, the top-right cyan, the bottom-left fuschia, and the bottom-right lime green.

EDU 007 PDE X, 2013, Manindra Moharana, Mentored by Daniel Shiffman.



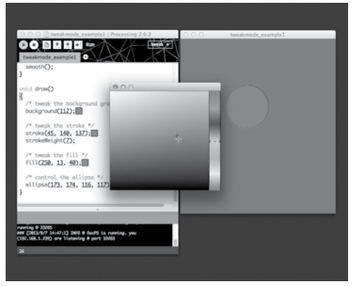

A cropped view of code written in the Processing text editor. A pop-up box that renames the variable “futureLocation” to “FL” appears over the editor.

PDE X, a continuation of Manindra’s 2012 GSOC work, is a Processing mode that augments the development environment with new features such as code completion, refactoring, real-time error checking, and debugging.

EDU 008 Serial Library for Processing, 2013, Gottfried Haider, Mentored by Casey Reas.

Serial Library replaces Processing's existing RXTX-based serial library with one built on top of Java Simple Serial Connector.

EDU 009 Tweak Mode, 2013, Gal Sasson, Mentored by Daniel Shiffman.



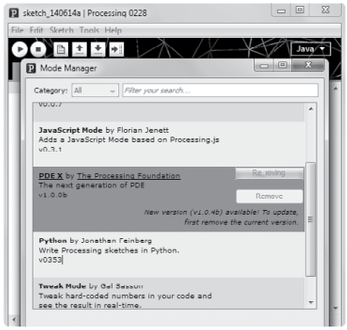
The Processing text editor is opened on the left and the respective sketch is displayed on the right. The sketch is of a red ellipse with a cyan stroke in the middle of a dark grey canvas. Example draw() code for an ellipse is written in the text editor, such as stroke() and fill(), with hard-coded numbers. A color picker appears over the code, corresponding to the ellipse's stroke() color.

TweakMode is a Processing mode that allows "tweaking" of hard-coded numbers in a sketch's code through a user interface.

EDU 010 Android Mode for Processing 3.0, 2014, Imil Ziyaztdinov, Mentored by Andres Colubri.

The new Android mode in Processing 3.0 implements several pieces of functionality that were missing from earlier versions: Export Signed Package (with transparent handling of keystores), device selector, automatic SDK download/installation, and target SDK selector.

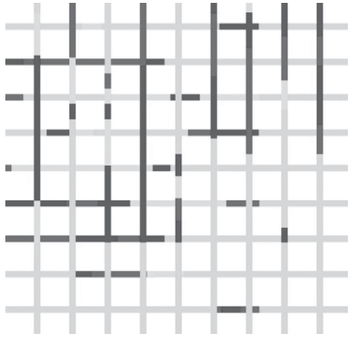
EDU 011 Contributions Manager: Reloaded, 2014, Joel Moniz, Mentored by Florian Jenett.




The Mode Manager is opened over the Processing text editor with several contributions listed within it. One of the contributions, titled "PDE X by The Processing Foundation," is highlighted. The "Remove" button next to the contribution title has been pressed, and a progress bar of its removal is displayed above the button.

This project added new features to the Contributions Manager, such as the addition, removal, and update of Tools and Modes without a restart, a new "examples-package"-type contribution, and highlighting contributions.

EDU 012 Loom, 2014, Cora Johnson-Roberson, Mentored by R. Luke DuBois.



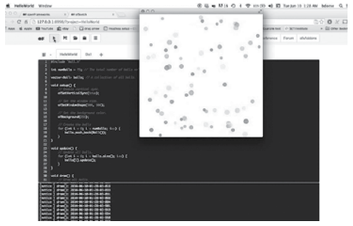
A white 10x10 grid with a thick multicolor stroke around each cell. Random portions of the stroke are either red, yellow, blue, or grey.

Loom lets you create and manipulate patterns of timed events, which could be mapped to audiovisual output, transformed in various ways, and recorded to enable non-real-time synthesis and synchronized video.

EDU 013 New Video Library Using GStreamer 1.x , 2014, Roland Elek, Mentored by Levente Farkas and Andres Colubri.

The aim of this project was to create a set of Java bindings for the GStreamer 1.x series, automatically generating everything where applicable, and then updating the video library in Processing to use these new bindings.

EDU 014 ofSketch, 2014, Brannon Dorsey, Mentored by Christopher Baker.



The browser-based text editor for openFrameworks is opened, and its respective sketch window appears on top of it. The sketch is of ellipses at varying sizes and yellow-orange hues randomly scattered throughout the canvas.

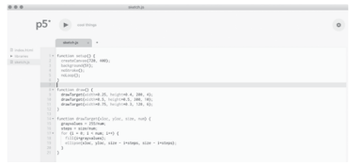
ofSketch is a barebones browser-based IDE for openFrameworks targeted towards new users.

EDU 015 p5.sound, 2014, Jason Sigal, Mentored by Evelyn Eastmond.

p5.sound brings the Processing approach to Web Audio, including functionality such as audio input, playback, manipulation, effects, recording, analysis, and synthesis with syntax built off of Wilm Thoben's Sound for Processing library.


A screenshot of the p5.sound library reference on the p5.js site.

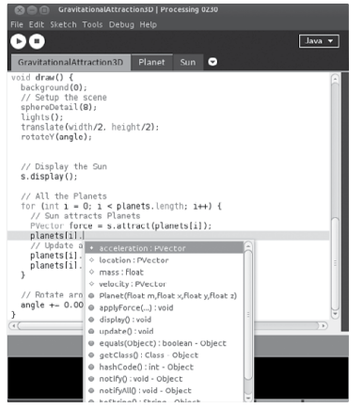
EDU 016 p5 IDE, 2014, Sam Lavigne, Mentored by Lauren Lee McCarthy.



The p5 text editor.

An easy-to-use desktop IDE for creating p5.js projects.

EDU 017 PDE X for Processing 3.0, 2014, Manindra Mohanara, Mentored by Daniel Shiffman.




The Processing text editor with some example draw() code. A pop-up box with a dropdown list of suggested properties and functions appears next to an unfinished line of code.

This project brought PDE X to a stable state and made it the default editor for Processing 3.0.

EDU 018 POculus, 2014, Pratik Sharma, Mentored by Elie Zananiri.

POculus provides an Oculus renderer for Processing.

EDU 019 Sound for Processing 3.0, 2014, Wilm Thoben, Mentored by Casey Reas.

New features, bug fixes, and cross-platform support were introduced to the Sound library for Processing.

EDU 020 TweakMode for Processing 3.0, 2014, Gal Sasson, Mentored by Christopher Baker.

Modifications and fixes were brought to TweakMode, a new execution mode in Processing 3.0 that allows changing sketch parameters in real time.

EDU 021 p5.js Windows Port, 2015, Guy de Bree, Mentored by Sam Lavigne.


A screenshot from the p5.js site of the p5 Editor download options for Mac OS X and Windows.

Developmen of a Windows port for the p5.js IDE and fixed bugs for said editor.

EDU 022 New Kinect Libraries openKinect and KinectPV2, 2015, Thomas Sanchez Lengeling, Mentored by Elie Zananiri.

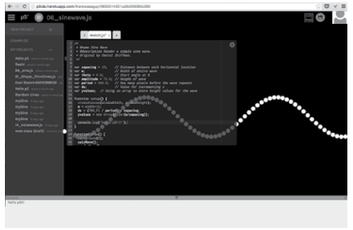
The goal of the project was to expand and update the existing Kinect libraries; OpenKinect-for-Processing and KinectPV2.

A grid of photos that demonstrates KinectPV2 functionalities.

EDU 023 Processing.py 3.0 Update, 2015, Luca Damasco, Mentored by Golan Levin.

Python Mode was revitalized and updated as users were now able to write Processing 3.0 sketches using Python syntax in the new editor.

EDU 024 Web IDE for p5.js, 2015, Jason Sigal, Mentored by Daniel Shiffman.

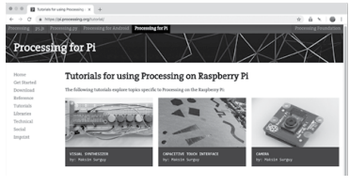


A screenshot of the browser-based code editor for p5.js.

A browser-based code editor was designed specifically for the p5.js community.

EDU 025 Raspberry Pi and armv7hf support, 2015, Gottfried Haider, Mentored by Ben Fry.

Gottfried worked on making Processing play well with the Raspberry




A screenshot of the webpage Processing for Pi showing the article for "Tutorials for using Processing and Raspberry Pi."

Pi and similar ARM-powered micro-computers (to borrow a term used in the 1970s to describe various kits and designs built around affordable microprocessors, such as the Motorola 6800).

EDU 026 p5.SVG and p5.PDF, 2015, Zeno Zeng, Mentored by Danne Woo.

A p5 sketch of an ellipse with the browser's inspector open. The SVG path of the ellipse is highlighted in the DOM.

The main goal of p5.SVG was to provide a SVG runtime for p5.js so that users could draw using p5's powerful API in SVG, save things to an SVG file, and manipulate existing SVG files without rasterization.

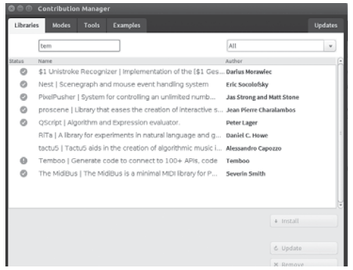
EDU 027 Internationalization of p5.js Website & Collection of Sketches for the p5.js Community Statement Video, 2015, Maya Man, Mentored by Johanna Hedva.

Maya Man worked primarily on two separate projects: 1) Rebuilt the p5.js site using Node.js, Assemble, Grunt, Handlebars, and YAML to support internationalization and 2) Curated sketches contributed from people around the world to play in the background of our p5.js community statement video.


A screenshot of the p5.js homepage with parts of text translated to Spanish. The top navigation reads "Descargar * Empezar * Referencia * Bibiliotecas * Aprender * Contribuir" in hot pink.

EDU 028 Contribution Manager UI upgrade, 2015, Akarshit Wal, Mentored by Scott Murray.

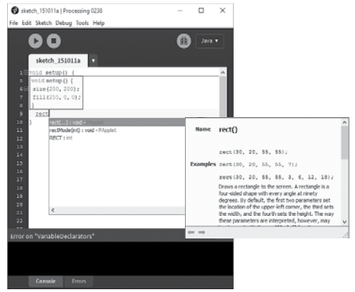
The aim of the project was to change the UI of the Contribution Manager and make it more user-friendly.




A screenshot of the Contribution Manager window showcasing the search and filter functionalities of the updated UI.

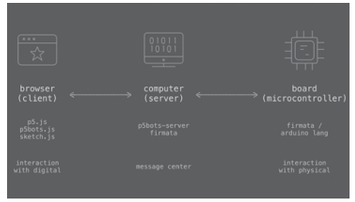
EDU 029 RSyntaxTextArea Integration and the REPL Mode, 2015, Joel Moniz, Mentored by Manindra Moharana.

The RSyntaxTextArea integration aimed at replacing the JEditTextArea, which was currently at the heart of the PDE editor, with the RSyntaxTextArea and the complementary AutoComplete code completion library to bring several new features to the table, such as code folding and documentation support and parameter tabbing during auto-completion.



A screenshot of the PDE after RSyntaxTextArea integration.

EDU 030 p5bots, 2015, Sarah Groff-Palermo, Mentored by Shawn van Every.



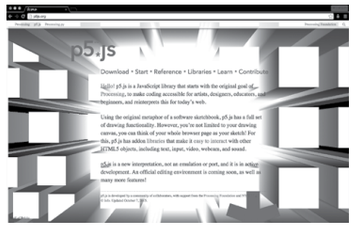

A flow that demonstrates how p5bots connects a microcontroller and the browser.

p5bots combines a socket layer and a simplified API to enable users to interact with an Arduino (or other microprocessor) from within the browser.

EDU 031 p5 WebGL Renderer, 2015, Karen Peng, Mentored by Kevin Siwoff.

A lightweight webGL renderer for p5.j was implemented, enabling users to create sketches under WebGL mode while all the syntaxes remain consistent with p5 2D (default) mode.





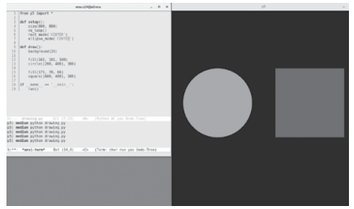

A screenshot of the p5.js homepage with a background image made with the webGL renderer.

EDU 032 Video and Audio Streaming Library, 2015, Nick Confrey, Mentored by Daniel Shiffman.

This project focused on streaming media (i.e. video and audio) over the network and revamped Processing's core video framework.

EDU 033 p5py, 2017, Abhik Pal, Mentored by Manindra Moharrana.

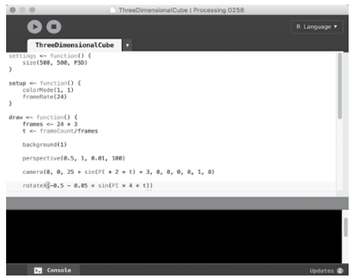
The main motivation behind creating p5py was to leverage Python's readability and Processing's emphasis on coding in a visual context to make programming easier to teach.



A screenshot of the web documentation for p5.py version 0.6.0.

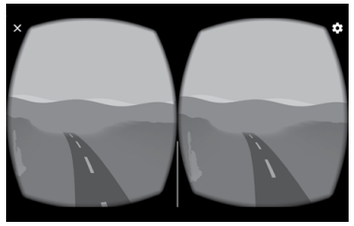
EDU 034 Processing.R, an Experimental New Mode in Processing for R Language, 2017, Ce Gao, Mentored by Jeremy Douglass.

Processing.R allows users to write Processing sketches using the R language.



The PDE after Processing.R is installed.

EDU 035 VR Audioscape, 2017, Sara Di Bartolomeo, Mentored by Andres Colubri, Gottfried Haider.



A screenshot of a landscape in VR Audioscape.

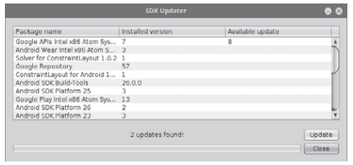
VR Audioscape is a virtual reality application made to demonstrate and document the possibilities of the new Processing Android Cardboard Mode.

EDU 036 Image Processing Library for Colorblindness, 2017, Sarjak Thakkar, Mentored by Claire Kearney-Volpe.

A photo of fruit processed with one of the correction algorithms, RGB Color Contrast Method.

An Image Processing Library was implemented to ease differentiation of colors for people with color-blindness.

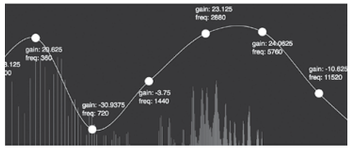
EDU 037 The New Processing for Android, 2017, Rupak Das, Mentored by Gottfried Haider.




The SDK Updater GUI displaying download progress.

Rupak Das transitioned the Processing for Android build process from using ANT scripts to Gradle, a newer build tool that was more compatible with the current Android SDK.

EDU 038 p5.Sound Developments, 2017, Jeevan Farias, Mentored by Jason Sigal.




A graph plotting the gain and frequency of a soundwave.

Jeevan Farias furthered development of the p5.Sound library, including new effects, presets, and modules for algorithmic composition.

EDU 039 Friendly Error System, 2017, Alice Chung, Mentored by Luisa Pereira.

Alice Chung expanded the p5.js Friendly Debugger, which checks function calls for correct parameter input, identifies common

A screenshot of the p5 editor demonstrating the Friendly Debugger in action.

JavaScript and p5.js errors, and provides feedback in a friendly and inclusive way.

EDU 040 p5.js 3D Rendering WebGL Mode, 2017, Kate Hollenbach and Stalgia Grigg, Mentored by Lauren Lee McCarthy, David Wicks, and Daniel Shiffman.

A screenshot of a custom shader rendering the surfaces of a grid of cubes. The shader blends an image and video using two different texture samplers in GLSL and tints the color according to mouseX and mouseY via custom uniforms.

Kate Hollenbach and Stalgia Grigg worked together to overhaul the 3D rendering WebGL mode to remove bugs, improve performance, and extend functionality.

EDU 041 P5.js Mappa Library, 2017, Cristóbal Valenzuela, Mentored by Daniel Shiffman.

Cristóbal Valenzuela built “Mappa,” a set of p5.js tools that facilitate the work between the <canvas> element and existing map libraries and APIs.

A map made with MapboxGL with p5 in WEBGL.

EDU 042 p5.js Release Process Automation and Library Modularization, 2017, Saksham Saxena, Mentored by Lauren Lee McCarthy.

Saksham Saxena worked on improving infrastructural aspects and operations of the p5.js library, independent of the library API itself.

EDU 043 p5.js en Español, 2017, Aarón Montoya-Moraga, Mentored by Lauren Lee McCarthy.


p5.js website switches language when clicking on the buttons EN and ES.

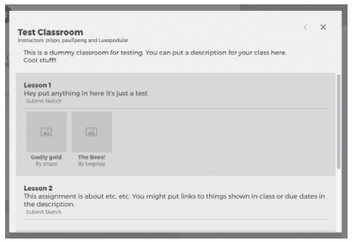
Aarón Montoya-Moraga worked on the development of the p5.js website and internationalization infrastructure in order to translate p5.js and Processing reference materials.

EDU 044 p5.js Web Editor Improvement, 2017, Jen Kagan, Mentored by Cassie Tarakajian.

Jen Kagan worked on improving the debugging and development experience in the p5 web editor by implementing autocomplete code suggestions, and improving the existing console.

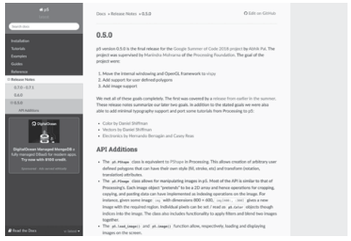
EDU 045 p5.js Classroom System, 2017, Zach Rispoli, Mentored by Cassie Tarakajian.

Zach Rispoli worked on a free-to-use classrooms system for p5.js, creating a system for instructors that are using p5.js to teach classes.




An early version of the classrooms system.

EDU 046 p5 (Python): Cross Platform Support, Image Support, and More, 2018, Abhik Pal, Mentored by Manindra Moharana.

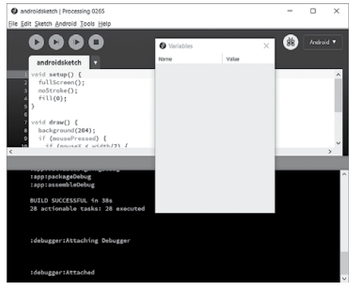


An early version of the classrooms system.

Abhik Pal continued the work on a native Python port of p5 from last year with the goal of fixing the cross-platform issues and further extending the API coverage.



EDU 047 Android Debugger: Processing-Android, 2018, Manav Jain, Mentored by Andrés Colubri and Rupak Das.




A window shows code for “androidsketch.”

A Debugger for the Android mode in the Processing Development Environment (PDE) was implemented.

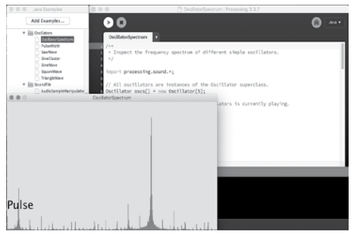
EDU 048 Implementing Missing WebGL Primitives in p5.js, 2018, Adil Rabbani, Mentored by Stalgia Grigg.

The project involved implementing missing WebGL primitives: arc, point, bezierVertex, curveVertex, quadraticVertex, and text in p5.js.


Three rendered figures on a white background. The title says “WebGL.” On the left is a graphic of a pizza labelled “Arc.” In the middle is a sphere made of points labelled “Point.” On the right is a daisy flower with orange petals, a stem, and a leaf, labelled, “Curves.”

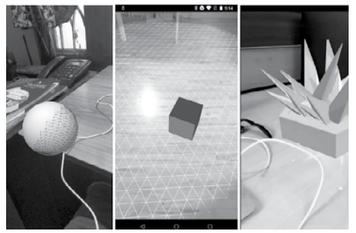
EDU 049 Processing Sound 2.0, Kevin Stadler, Mentored by Casey Reas.

Kevin Stadler successfully undertook a complete rewrite of Processing’s Sound library through a fully backwards-compatible re-implementation based on a Java synthesis engine and beginner-friendly example sketches.




A window of Processing code for a program named “OscillatorSpectrum” is shown in the background. In the foreground is the resulting program, which shows a window with a green background, the text “Pulse,” and at the bottom of the window, a sound graph is rendered in hot pink.

EDU 050 AR Renderer: Processing-Android, 2018, Syam Sundar K, Mentored by Andrés Colubri and Jesus Duran.

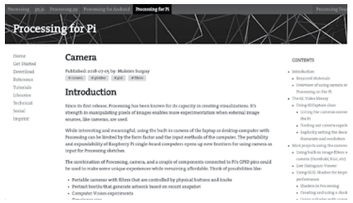


An image divided into three vertical sections. On the left, a 3D sphere is rendered in a photograph. In the middle a grid and a 3D red cube is rendered in a photograph. On the right, a 3D shape of spikes emerging from a rectangular cube is rendered in a photograph.

An ARcore Renderer was created which focuses on creating Augmented Reality applications using Processing – Android, that will be able to render 3D objects onto the real world scene using Processing code in real time.

EDU 051 Processing for Pi Website and Tutorials, 2018, Maksim Surguy, Mentored by Gottfried Haider.

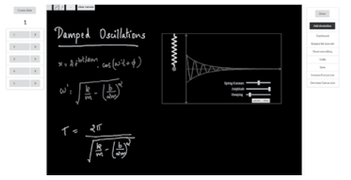
The result of this project was a website and a set of highly detailed tutorials for working with Processing on Raspberry Pi single board computers.




A screenshot of the webpage for Processing for Pi, showing the article for “Camera.”

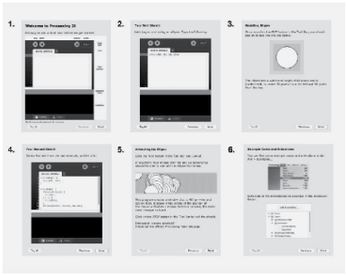
EDU 052 Dynamic Learning, 2018, Jithin KS, Mentored by Saber Khan.

The project involved development of a web app called Dynamic Learning, which is an online platform where STEM teachers and creative coders can collaborate to create lessons that include interactive visualizations created in p5.js.




Most of the frame is filled with a black square that has handwriting in different colors on it. The text reads “Damped Oscillations” and shows mathematical equations and diagrams. On the left side of the image is a list of the number of slides. This image shows that we are on the first slide. On the right side of the image is a menu with buttons that include: “Draw, Add Simulation, Dashboard, Request for new sim, Reset everything, Undo, Save, Increase Canvas Size, Decrease Canvas size.”

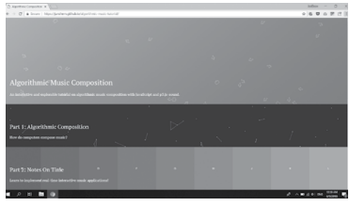
EDU 053 Development Environment: Beginner/New User Experience Features, 2018, Jae Hyun, Mentored by Elie Zananiri.



Six screenshots of the frames in the new Getting Started Tool tour. Each frame shows an image with text about a specific feature in Processing 3. The first six, shown above, include: Welcome to Processing 3!, Your First Sketch, Resulting Ellipse (which shows the resulting program from the previous frame), Your Second Sketch, Animating the Ellipse, and Example Code and Animations.

The project involved developing two contributed Tools for New/Beginner users: 1) the Getting Started Tool, which consists of several frames that give new users a short tour of the PDE and 2) the Reference Tool, which provides a built-in reference feature and eliminates the need for users to open up a browser to find a Processing Reference.

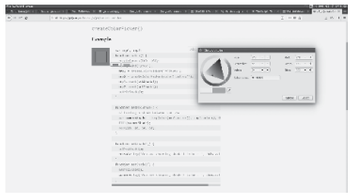
EDU 054 Beginner/New User Platform for Algorithmic Composition on p5.Sound, 2018, Chan Jun Shern, Mentored by Jason Sigal.




A screenshot of a browser window split into three parts, each with a different color and a few simple graphics made of lines. The top part reads, "Algorithmic Music Composition"; the middle part reads, "Part 1: Algorithmic Composition"; the bottom part reads, "Part 2: Notes on Time."

Chan made p5.Sound a friendly platform for algorithmic music composition tasks by building up features, fixing bugs, adding documentation, and producing examples of p5.js sketches related to algorithmic composition.

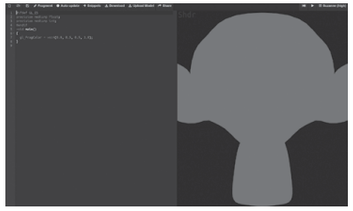
EDU 055 Improvements to I/O Methods for p5.js, 2018, Tanvi Kumar, Mentored by Alice M. Chung.



A new example for createColorPicker().

Tanvi Kumar improved the I/O methods of p5.js by resolving existing issues in I/O and testing methods and various file types on different browsers.

EDU 056 GLSL Editor for Processing, 2018, Izza Tariq, Mentored by Andrés Colubri.



A screenshot of the Shdr editor, with code on the left side of the window and the results on the right.

Izza Tariq developed a code-based (GLSL) shader editing tool for the Processing Development Environment (PDE) that updates the PDE sketch display window in real time without having to compile the code repeatedly.

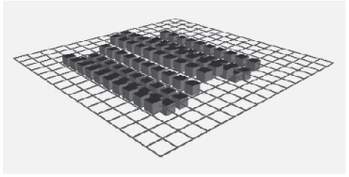
EDU 057 New JavaScript Console in p5.js Web Editor, 2018, Liang Tang, Mentored by Cassie Tarakajian.

Liang integrated a new console to p5.js web editor by improving the original console, implementing new features, and making the console interactive.

A p5.js sketch showing new console functions like console.log(), console.warn(), and console.debug().

EDU 058 Improvements to WebGL Mode in p5.js, 2018, Aidan Nelson, Mentored by Kate Hollenbach.

Aidan Nelson implemented a number of ease-of-use improvements to p5.js's WebGL (3D) mode, which allowed a beginner coder to visualize and understand 3D space.

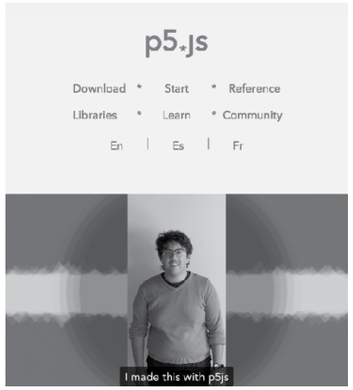


An animated GIF shows a grid of boxes move up and down in 3D space as our view rotates around the scene.

EDU 059 Updating hello.p5js.org, 2018, Elgin-Skye McLaren, Mentored by Evelyn Masso.

Elgin-Skye McLaren created a new interactive hello.p5js video

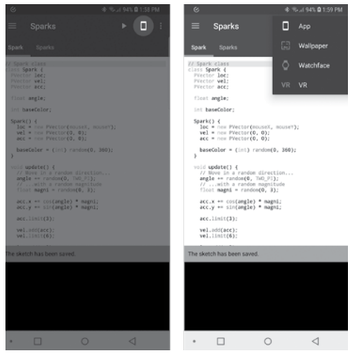





A screenshot that includes part of the p5.js browser. At the top, in hot pink text, is a menu that reads, "p5.js: Download * Start * Reference * Libraries * Learn * Community. en | es | fr." In the center is a video of a person with a graphic of scattered lines in a band of color; the caption at the bottom of the video reads, "I made this with p5js."

trailer and website to welcome new users to the p5 community.

EDU 060 APDE Beta Push, 2018, William Smith, Mentored by Sara Di Bartolomeo.

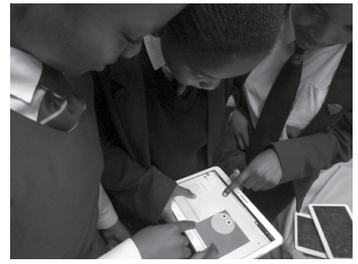



A screenshot of two ADPE windows showing code. Both are identical except the left window has an overlay to highlight the target selection button. The right window has no overlay but shows the target selection dropdown when the button is pressed.

APDE (Android Processing Development Environment) is a fully functional IDE for creating Processing sketches on Android devices, but was in need of improvements to keep it up to date with the desktop version of Processing and to improve its accessibility to new users.

EDU 061 Test Strategy for Maintaining and Updating Mobile Functionality of p5.js, 2018, Sithe Ncube, Mentored by Lee Tusman.

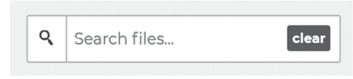
The main goal of the project was to design a test strategy and extensively test and update the p5.js mobile functionality so that compatibility issues can be tracked easily with updates to the library and mobile platforms.



Three children huddle around and point to a tablet which shows a p5.js sketch.

EDU 062 Search Bar for Sketches in the p5.js Web Editor, 2019, Rachel Lim, Mentored by Cassie Tarakajian.

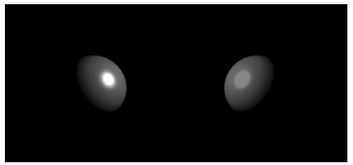
This project provided more convenience for organizing and retrieving sketches within an individual account through a search bar and a collections tab.




The search bar in an active state.

EDU 063 Advancing p5.js's WebGL Mode, 2019, Sanket Singh, Mentored by Adam Ferriss.

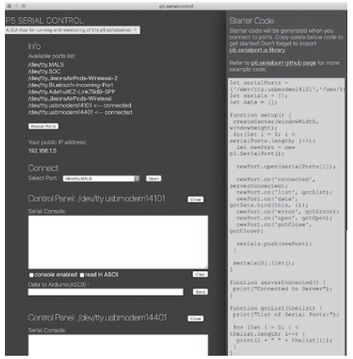
This project implemented new functionalities for p5.js using WebGL to expand the current functionality related to lighting.



Two black spheres in front of a black background. The scene contains a red light. The left sphere is shining with white light, whereas the right sphere is shining with a red light.

EDU 064 Updating and Improving p5.Serial, 2019, Jiwon Shin, Mentored by Shawn Van Every.

Jiwon Shin updated and made improvements to the functionality of the p5.serial library, a commonly used library to connect serial devices to p5.js sketches.

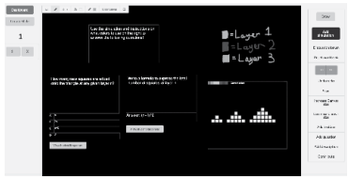



A screenshot of the application window for the latest release of p5.serialcontrol in the new grey and yellow color scheme.

EDU 065 Visualizing STEM Education with Dynamic Learning, 2019, Ashneel Das, Mentored by Nick McIntyre.

This project improved and extended Dynamic Learning by focusing on

three major areas of extension: interface changes and responsiveness, integration with other software, and classroom usability improvements.



A screenshot of a black work area with white text, includes a textbox, two questions, and a simulation about patterns of growth.

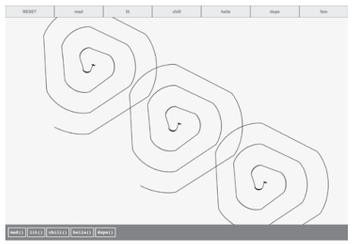
EDU 066 Math in Motion, 2019, Alexandra Cheng and Oskar Garcia, Mentored by Greg Benedis-Grab and Ellen Nickles.

Math in Motion (MiM) is a modern interface for working with math on the web.


"p5 MiM" in large, hot pink text.

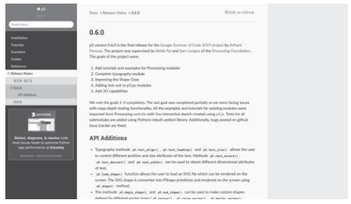
EDU 067 Code Slang, 2019, Jenna Xu, Mentored by Sharon Lee De La Cruz.

Jenna Xu helped develop a JavaScript library that is flexible, intuitive and human; whose syntax resembles natural language more than programming language; and that transforms programming into a fun conversation with the computer rather than a rigid set of logical commands.



A screenshot of the Code Slang interface featuring a sketch made up of red lines in the middle. The buttons at the top read "mad," "lit," "chill," "hella," "dope," and "fam."

EDU 068 Completing p5.py API and Improving Documentation, 2019, Arihant Parsoya, Mentored by Sam Lavigne and Abhik Pal.



A screenshot of the web documentation for p5.py version 0.6.0.

The aim of this project was to make p5.py ready for public use by completing the APIs to make it on par with Processing and p5.js.

EDU 069 Curating Community Creativity for p5.js 1.0, 2019, Ashley Kang, Mentored by Kate Hollenbach.

Ashley Kang curated six projects from the online and offline p5.js community and created a place for them on p5js.org.

EDU 070 Improving p5.js Unit Tests, 2019, Urvashi Verma, Mentored by Evelyn Masso.

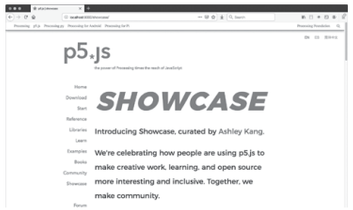
This project focused on improving the p5.js unit tests, creating a more complete unit test coverage, and implementing tutorials for new contributors.

EDU 071 Maintenance of Android Mode: SDK Downloader/Updater, Emulator, Library Structure, 2019, Deeraj Esvar R, Mentored by Sarah Lensing and Cristian Mosquera.

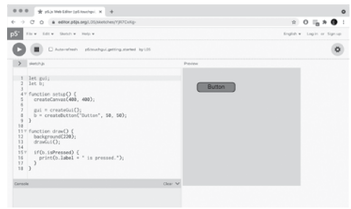
This project focused on implementing an up-to-date SDK and Emulator installer to improve its functionalities.

EDU 072 p5.touchgui, 2019, Carlos L05 Garcia, Mentored by Yining Shi.

p5.touchgui makes it easy to add buttons, sliders, and other GUI (graphical user interface) objects to p5.js sketches, enabling users to focus on quickly iterating ideas with easily created GUI objects that work with both mouse and multi-touch input.




A screenshot of the new Showcase page on p5js.org.

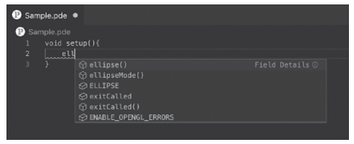


A screenshot of the p5 Editor showing code on the left and the results on the right.



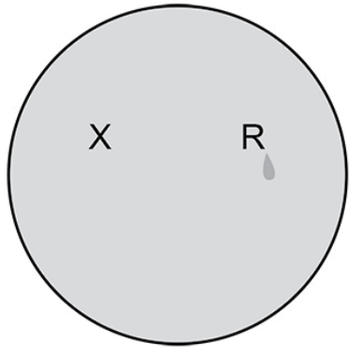
EDU 073 Processing Language Server, 2019, Syam Sundar K, Mentored by Manindra Moharana.

Processing Language Server focuses on creating a Language Server Protocol implementation for Processing Programming Language.



A screenshot of Processing code written in VScode with auto-completion.

EDU 074 Stabilizing and Improving p5.xr During Alpha Release, 2019, Vedhant Agarwal, Mentored by Stalgia Grigg.




A yellow circle with a black stroke with the letters "X" and "R" within it. A teardrop is underneath the "R."

p5.xr is a library for p5.js that enables WebXR capabilities with p5 sketches.

EDU 075 Stabilizing Processing Video with GStreamer 1.x, 2019, Alex Stamm, Mentored by Andres Colubri.

The goal of this project was to stabilize the Processing Video Library to v2.0, which is based on the GStreamer media framework.

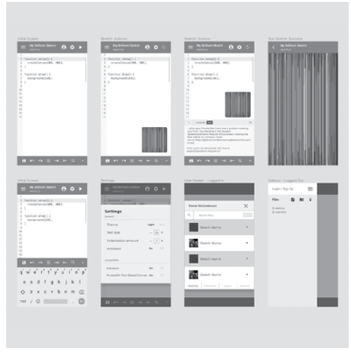
A screenshot of a window playing a video of a bee on a leaf.

EDU 076 Using Audio Workley in the p5.Sound Library, 2019, Oren Shoham, Mentored by Jason Sigal.

This project added AudioWorklet support to p5.Sound, allowing certain parts of the library to run more efficiently by moving custom audio processing to a separate audio thread.

EDU 077 Improved Web Editor Mobile UI, 2020, Ghales Trilo, Mentored by Cassie Tarakajian.

Ghales Trilo worked on improving the experience of the web editor on phones and tablets by detecting, validating, studying, and providing solutions for weak points on the mobile and tablet experience.



First design draft of the mobile layout.

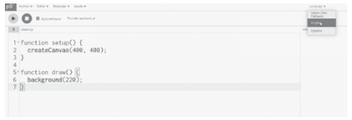
EDU 078 p5 for 50+ Teaching, 2020, Inhwa Yeom, Mentored by Qianqian Ye.

Inhwa Yeom reached out to educators around the world, who are aged 50 years old and up, with the aim to contribute to documenting, showcasing, and sharing teaching experiences, specifically by re-using the existing features of p5js.org.


In-progress website with title "p5 for 50+ aims to enhance digital literacy and rights of people going on 50+."

EDU 079 P5 Web Editor Spanish Translation, 2020, Omar Verduga, Mentored by Andrew Nicolaou.

Omar Verduga improved and expanded the Spanish translation of the p5.js web editor and documentation.



A screenshot of the different menu options in the web editor, that are switching between English and Spanish.

EDU 080 Extending p5.js Friendly Error System, 2020, Akshay Padte, Mentored by Stalgia Grigg.


A log of error messages from the Friendly Error System.

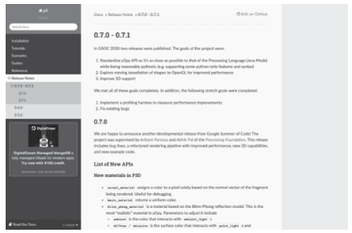
Akshay Padte made p5.js's Friendly Error System even friendlier by adding new features and fixing existing issues.

EDU 081 p5.js Accessibility and Canvas Descriptions, 2020, Luis Morales-Navarro, Mentored by Kate Hollenbach.

Luis Morales-Navarro made p5.js more accessible by creating a describe() function that allows users to write their own text-based canvas descriptions.

EDU 082 Improving p5.py, 2020, Ziyao (Mark) Zhang, Mentored by Arihant Parsoya and Abhik Palxz.

Ziyao (Mark) Zhang worked to improve p5.py, including standardizing its API, and implementing some new features in 3D mode.



A screenshot of the web documentation for p5.py version 0.7.0 - 0.7.1.

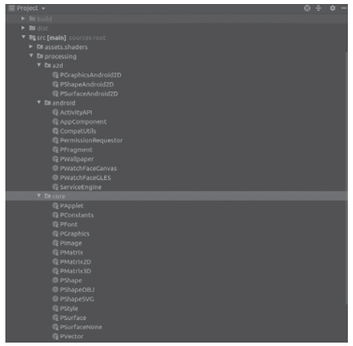
EDU 083 Swift Processing Library Development, 2020, Juan Lee, Mentored by Jonathan Kaufman.

A rotating group of basic shapes ranging from a cube to a sphere, run on the 3D graphic module in Swift Processing.

Juan Lee worked to bring the camera functionality and 3D primitives to the Swift Processing Library.

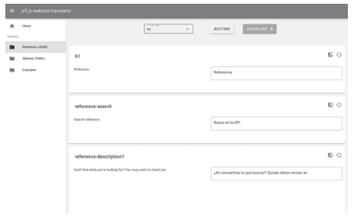
EDU 084 Support for Kotlin Native, 2020, Aditya Rana, Mentored by Syam Sundar K.

Aditya Rana worked on the migration of Android-Processing mode and Migrating Groovy based Gradle System to Kotlin, and implementing the multiplatform library in iOS for as many stubbed methods as possible.



A panel that lists multiple folder structures.

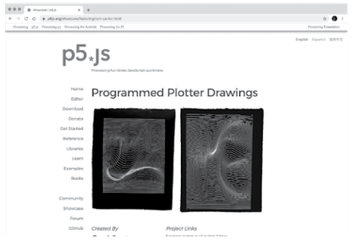
EDU 085 Italian Translation and i18n Improvements, 2020, Yukie Nomiya, Mentored by Evelyn Masso.



The p5.js-website-translator UI.

Yukie Nomiya made the internationalization process of the p5.js website easier and more accessible to contributors, while also simplifying the maintenance of the translation.

EDU 086 Increasing Organization and Scope of the p5.js Showcase, 2020, Connie Liu, Mentored by Joey Lee and Yining Shi.




p5.js website with page titled "Programmed Plotter Drawings" and two images of plotter drawings in green and white.

Connie Liu expanded the p5.js showcase to allow users to be able to easily search for inspiration. They could specify projects that use a specific library or function. The aim was to inspire users by the great diversity in p5.js's community.

EDU 087 Use ES6 Import and Classes in p5.Sound Library, 2020, Divyanshu Raj, Mentored by Kyle James and Jason Sigal.

Divyanshu Raj worked with the p5.sound library to upgrade the codebase to use ES6 features and change the current module format (current AMD) and loaders (requireJS) used in the module system.

EDU 088 Addon Library Development—p5.teach.js, 2021, Aditya Siddheshwar, Mentored by Nick McIntyre and Jithin KS.

Aditya Siddheshwar worked on p5.teach.js, which involved developing tools for teaching STEM through p5.js, adding functions to animate shapes, and animating math symbols. The library provides educators tools to make interactive animated sketches that support learning in the remote environment.

EDU 089 Improving the p5.xr Library Through Artistic Examples, 2021, Anais Gonzalez, Mentored by Stalgia Grigg.

Anais Gonzalez created a series of interactive examples for the p5.xr library that showed people how to work with creative coding concepts inside of a virtual space.

EDU 090 Korean Translations and Website Improvements, 2021, Joseph Hong, Mentored by Jiwon Shin.

Joseph Hong worked on updating and adding to the Korean translations of the p5.js website. He also restructured navigation within the site to improve the user experience of those new to p5.js.

EDU 091 p5.js 2021 Showcase: The Love Ethic, 2021, Katie Chan, Mentored by KT Duffy and Sam Vassor.

Katie Chan worked on the third iteration of the p5.js showcase this summer. Unlike previous years, this showcase had a theme: "The Love Ethic," which was heavily inspired by author bell hooks and her writing on embracing love in all aspects of our life. This showcase had a particular focus on the softness of the programming language and the ability of p5.js to be a radical community with values rooted in social justice and accessibility.

EDU 092 Adding Alt Text, 2021, Ka tie Liu, Mentored by Rachel Lim.

Katie Liu focused on adding alt text to visual elements on the p5.js website to increase the website's accessibility for all users. The addition of alt text helped users who use screen readers to better understand tutorials and examples on the p5.js website.

EDU 093 Add Examples and Fix Bugs in Swift Processing, 2021, Masood Kamandy, Mentored by Jon Kaufman.

Masood Kamandy continued developing the Swift Processing library. His focus will be adding live-coding educational playgrounds to the library via Xcode Playgrounds and SwiftPlaygrounds for the iPad. He also helped with the iOS 15 version of the library and identified bugs/fixes during the development of the educational materials.

EDU 094 Activism Through Storytelling with Code, 2021, Niki Ito, Mentored by Elgin-Skye McLaren.

Niki Ito created a web-based guide for artists, designers, and coders to learn potential collaborations between art, community, and code. The guide followed her project, "Activism Through Storytelling with Code" as an example. The project was an interactive code-based website with visual narratives that illustrate conversations documented within the Asian community. The website shed light on their experiences and created a stronger bond between individuals and cultures.

EDU 095 Improve Test Coverage in p5.Sound Library, 2021, Sai Bhushan, Mentored by Guillermo Montecinos.

Sai Bhushan improved the test coverage and the testing architecture of the p5.Sound library.

EDU 096 Hindi Translation for p5.js Website, 2021, Sanjay Singh Rajpoot, Mentored by Aditya Rana.

This project added a Hindi language translation and better language support added to its bucket. Sanjay Singh Rajpoot also improved the i18n engine for better translation across all languages. With proper implementation of the new functionality, the site will be future-proof and stable for the up-coming versions of p5.js.

EDU 097 New Contributor Onboarding Project, 2021, Ashley Jane Lewis, Mentored by Sarah Ciston and Kenneth Lim.

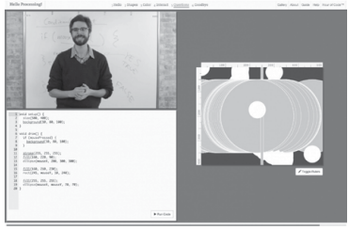
This project involved thinking about the experience for new contributors to the project with the guiding question: Wow can the README, developer_docs, and other documentation be improved to make it easier for new people to contribute, especially those without deep experience coding or contributing to open source?

EDU 098 Friendly Error System Project, 2021, Alice Chung,

Mentored by Aren Davey and Kate Hollenbach. This project involved improving the Friendly Error System (FES) documentation by updating the FES developer doc, adding comments inline to the code, and implementing internationalization/localization to the FES so error messages may be given in different languages.

EDU 099 code.org, 2013–2017.

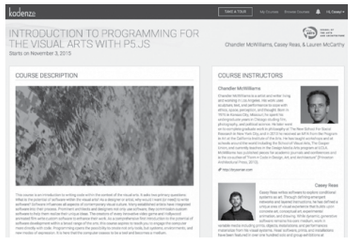

A screenshot of the Processing section on the code.org website. The webpage reads "Hello Processing" with circles of different hues and opacities behind the text.



A screenshot of a Processing tutorial on the code.org website. The screen is split into three tiles: the top-left tile is of a person in glasses standing in front of a whiteboard, the bottom-left is a code editor displaying some Processing code, and the right screen is the visual output of the code.

In partnership with code.org and their annual, global Hour of Code event, the Processing Foundation made one hour of code tutorials in online video form called Hello Processing. The tutorials are geared toward middle and high school students to learn Processing. The tutorials are distinct in their emphasis on creativity and visual art in programming, focusing on drawing, sketching, and visual interaction.

EDU 100 Kadenze, 2015.




A screenshot of the "Introduction to Programming for the Visual Arts with p5.js" section from the Kadenze website. There are two modules titled "Course Description" and "Course Instructors."

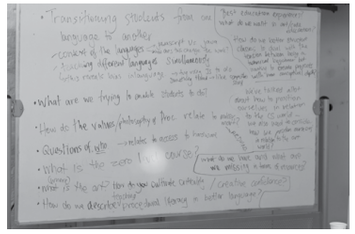
Teachers stand in front of green screen, their image mirrored on monitor nearby.

The Processing Foundation partnered with Kadenze to make a series of free, online classes around Processing and p5.js. Courses included "Introduction to Programming for the Visual Arts with p5.js," taught by Lauren Lee McCarthy, Casey Reas, and Chandler McWilliams and "The Nature of Code," taught by Daniel Shiffman.

EDU 101 Learning to Teach, Teaching to Learn, 2016–2021.

Organizers stand in front of a class.

People sitting at a desk in discussion.



A whiteboard full of ideas, questions, thoughts, and brainstorms.

People sitting at rows of tables in conversation.

People sitting in a socratic circle in discussion.



Learning to Teach, Teaching to Learn is a series of events for educators who teach computer programming in creative and artistic contexts. Founded by the School for Poetic Computation in partnership with the Processing Foundation and directed by Tega Brain and Taeyoon Choi, the events bring together experienced educators to explore pedagogy, curriculum development, and how to create environments and tools for learning, specifically for interdisciplinary art practices.

EDU 102 Learning to Teach Creative Technologies Remotely, January 22–23, 2021, Hosted by Integrated Digital Media at NYU Tandon School of Engineering, Organized by De Angela L. Duff, Tega Brain, R. Luke DuBois, Reginé Gilbert, Kathleen M. McDermott, and Ashley Jane Lewis.

Learning to Teach Creative Technologies Remotely: A Virtual UnSymposium was an opportunity to participate in conversations on how to foster creativity and experimentation within this new teaching and learning landscape. In an UnSymposium format, we got together and shared what worked and didn't work in our remote and hybrid classrooms this past. We discussed workshop opportunities and challenges for our pedagogy in the coming semester.

Session topics included creative coding, physical computing, and experimental realities (AR/VR/XR or motion capture). Questions concerning what's worked and what hasn't worked with Zoom instruction came up in a session called "Beyond Zoom: Getting Weird with Teaching Online," as well as questions about hosting virtual events and best practices for fostering student or colleague virtual engagement.

EDU 103 Learning to Teach, Teaching to Learn, January 20, 2019, Hosted by UCLA Design Media Arts.

Learning to Teach is a daylong conference for educators teaching computation in creative fields like art, design, or digital humanities departments. Led by Taeyoon Choi (SFPC) and Tega Brain (NYU), the the conference featured keynotes by leading educators including Elise Co (Art Center), Rhazes Spell (UCLA), and Lauren Gardner (SFPC). The speakers shared their experiences teaching computer programming, design, game, and related topics; their strategies for blending critical thinking and engineering; and inspiring creativity in a teaching environment. This program also included a participatory session for attendees to observe the pedagogical strategies of their peers and develop teaching approaches of their own.

EDU 104 Learning to Teach, Teaching to Learn, January 20, 2018, Hosted by NYU Integrated Digital Media, Brooklyn, Presented by Daniel Shiffman, Naomi Clark, and Brad Garton.

The event consisted of a one-day program of speakers followed by a one-day open workshop for developing one's own syllabus and course materials.

EDU 105 Learning to Teach, Teaching to Learn, Houston, December 15, 2017, Hosted by Moody Center for the Arts, Rice University, Houston, Organized by Tega Brain, Taeyoon Choi, and Lauren Lee McCarthy in collaboration with UCLA Conditional Studio.

The event consisted of a one day program of speakers followed by a one-day open workshop for developing one's own syllabus and course materials.

EDU 106 Learning to Teach, Teaching to Learn, 2017, Hosted by Postlight, NYC, Presented by Lauren McCarthy (UCLA DMA), Kaho Abe (Eyebeam Art and Technology Center), Mimi Yin (NYU ITP), and Nick Montfort (MIT).

The second edition of the Learning to Teach mini-conference was held on January 15, 2017 in New York City. Hosted by the School for Poetic Computation and in partnership with the Processing Foundation, this daylong conference was an open forum for educators teaching computer programming in creative and artistic contexts. The morning session consisted of a series of talks by experienced educators who shared their teaching philosophy, teaching strategies, and approaches to assessment and student feedback. In the afternoon, participants were invited to workshop sessions to discuss curriculum development, pedagogy, and tools for learning. This conference explored the intersection of pedagogy and

creative practice, and provided an opportunity to share ideas for another year of teaching ahead.

EDU 107 Learning to Teach at ICP, November 30, 2016, Hosted by International Center of Photography, NYC, Presented by Tega Brain, De Angela L. Duff, Aankit Patel, and "BBQ Dave" Sheinkopf.

The event consisted of a one-day program of speakers followed by a one-day open workshop for developing one's own syllabus and course materials.

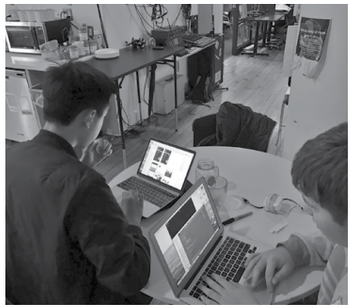
EDU 108 Learning to Teach, Teaching to Learn, January 9-10, 2016, Hosted by School for Poetic Computation, NYC, Presented by Katherine Bennett, Golan Levin, Zach Lieberman, Stacey Mulcahy, Allison Parrish, and Dan Shiffman.

This was a two-day conference for educators teaching computer programming in creative and artistic contexts. The event focused on asking experienced educators: How can we learn to teach more effectively? It brought teachers together to explore pedagogy, curriculum development, and strategies for creating environments and tools for learning. The event consisted of a one-day program of speakers followed by a one-day open workshop for developing one's own syllabus and course materials. It also included a session on developing guidelines for teaching creative computation and exploring how to best publish syllabus resources online.

EDU 109 Creative Coding Fest, Oct 22, 2016, Hosted by New York City (NYU ITP).


A vaporwave-inspired flyer that's titled "NYU ITP CC Fest" with smaller text underneath that reads "Creative Coding w/ p5.js."

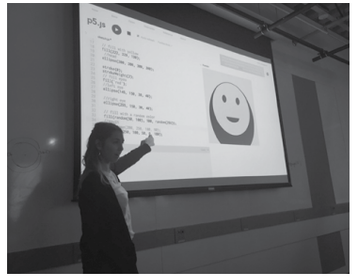
Two speakers stand in front of a crowded classroom with their presentation projected onto two screens.



Two students sit at a roundtable with their laptops open.

On a rainy day in NYC, forty students and teachers gathered together at NYU ITP to spend the day learning how to use p5.js and thinking about how to use it in the classroom. Keynote speakers and workshop leaders included Saber Khan, Dan Shiffman, Lauren Lee McCarthy, Danny Fenjves, Stephen Lewis, Sinan Ascioglu, Marius Watz, and Katy Garnier.

EDU 110 Creative Coding Fest, Apr 23, 2017, Hosted by New York City (NYU MAGNET).



A person stands in front of a class pointing at a projection of the p5 editor.

A teacher stands in front of two rows of students with a projection of the p5 editor.

Keynote speakers and workshop leaders included Dan Shiffman, Cassie Tarakajian, Sinan Ascioglu, Sofia Garcia, and more.

EDU 111 Creative Coding Fest, September 23, 2017, Hosted by Los Angeles (UCLA DMA).

Lauren at the podium presenting in front of an audience. The presentation is projected onto a large screen.



Three tables filled with students sitting at their iMacs. The teacher stands at the podium to the left.

An aerial view of three young students at their laptops with the p5 editor opened.

A table filled with students sitting at their iMacs.

CC Fest made its way to Los Angeles with keynote speakers and workshop leaders including Lauren Lee McCarthy, Dan Shiffman, Cassie Tarakajian, and more.

EDU 112 Creative Coding Fest, November 12, 2017, Hosted by New York City (NYU ITP).

Keynote speakers and workshop leaders included Dan Shiffman, Cassie Tarakajian, Jeff Olson, Serena Parr, and more.

A person sitting at the front of the roundtable with students in front of a presentation projected onto the wall behind them titled "Creative Live Communications."

Three people sitting at a desk and looking at the laptop in the middle.

Two people in mid-conversation. There's a crowd behind them and the walls are exposed brick.

A person stands in front of a crowded classroom with a presentation projected on the wall behind them. The presentation slide has the Glitch logo on the left with the text "Remix this p5.js starter project" underneath, and the p5.js logo on the right with the text "p5js.org references & examples" underneath.

Three individuals are looking at a screen from a distance, the screen being out of frame. One of the individuals is pointing in the screen's direction.

EDU 113 Creative Coding Fest, May 5, 2018, Hosted by New York City (NYU MAGNET).

Keynote speakers and workshop leaders included Jenn Schiffer, Ari Melenciano, Yining Shi, Cristobal Valenzuela, Jeff Kaufman, Liam Baum, Zach Brewer, Matt Carlberg, Tega Brain, Justin Gohde, Georgio Nicolas, Leandra Tejedor, Sinan Ascioglu, Danny Fenjves, and more.

A group of people of different ages are sitting at individual desks with laptops.

A person with a laptop looking up at the projection screen in a classroom. There is a group of people sitting behind her in desks.

A person standing and reaching over to the keyboard on a laptop. Two children sitting look on.

EDU 114 Creative Coding Fest, October 13, 2018, Hosted by San Francisco (SF Friends School).

CC Fest made its debut in the Bay Area, as students and teachers spent the day using p5.js, Processing, and Arduino. Keynote speakers and workshop leaders included Kelly Lougheed, Xiaohan Zhang, Amy Wibowow, Lark Alder, and more.

A group of forty people facing the camera and smiling.

A person pointing at a large screen displaying a presentation.

A person crouching and reaching over to the keyboard on a laptop. A child looks on.


A person on the podium speaking. Their presentation is projected onto the screen behind them.

EDU 115 Creative Coding Fest, November 10, 2018, Hosted by New York City (NYU ITP).

Two people with laptops looking at their screens.

An older individual and a child looking at a laptop.

Keynote speakers and workshop leaders included Nicole He, Ingrid Burrington, Sofia Garcia, Jeff and Taylor from Upperline Code, Stepegen Lewis, Katy Garnier, Liam Baum, Colleen Lewis, and more.

A crowded classroom of students sitting at a table with their individual laptops. There's a workshop leader standing in front of the class with their presentation projected behind them. The background is a little out of focus.

Students of different ages sitting at tables with individual laptops.

Four children crowding around a laptop. An older individual is also looking over.

EDU 116 Creative Coding Fest, June 8, 2019, Hosted by New York City (NYU MAGNET).

A person standing in front of a classroom with a code editor projected onto the screen behind them.



A group of around 55 people facing the camera and smiling.

A person crouching between two people looking at a laptop.

A laptop on a desk the webcam video is displayed on the screen. A person with one hand up. Green dots and boxes mark different parts of the arm and hand. A photographer and their camera are next to the person with their arm raised.

People sit at tables, each with their individual laptops, as they look at someone writing on a whiteboard.

Keynote speakers and workshop leaders included Maya Man, Nabil Hassein, Celeste Layne, Liam Baum, Tim Chen, and more.

EDU 117 Creative Coding Fest, October 19, 2019, Hosted by San Francisco (SF Friends School).

Five young students sitting around the table with their laptops with an older individual.

A group of about forty people sitting in rows of chairs and looking at the camera smiling.

Keynote speakers and workshop leaders included Stevon Cook, Monica Dinculescu, Kevin Workman, Chris Barrious, Saskia Leggett, Angi Chau, Greg Beutler, Art Simon, Radamés Ajna, Suyash Joshi, and Rushali Paratey.

EDU 118 Creative Coding Fest, October 27, 2019, Hosted by Los Angeles (Crossroads School).

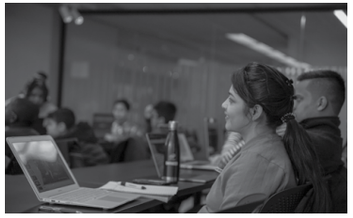

A group of about thirty people looking at the camera smiling. An image is projected onto the screen behind them that displays “Crossroads School for Arts & Sciences” and the Processing logo underneath.

Keynote speakers and workshop leaders included Jo Wright, Evelyn Masso, J.D. DeVaughn-Brown, Kelly Lougheed, Kevin Workman, Paul Way, and Scott Gruber.

EDU 119 Creative Coding Fest, December 8, 2019, Hosted by New York City (NYU ITP).

Keynote speakers and workshop leaders included Aankit Patel, Chancey Fleet, Liam Baum, Kyle James, Jeff Olson, Esther Hersh, Luca Damasco, and more.

A group of people of different ages sitting at tables with their individual laptops. They are all looking to the left.



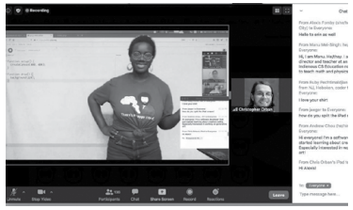
A row of people sitting with their laptops. The person in the foreground is smiling as they look to the front of the classroom.

A person sitting at a desk with their presentation projected on the screen behind them.

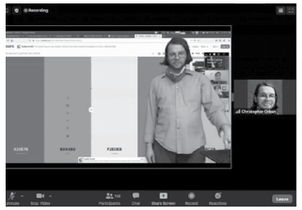
A young person sitting at a desk and looking into their laptop.

Three young individuals sitting at a desk with their laptops. An adult is crouching and reaching over to the keyboard of one of the laptops.

EDU 120 Creative Coding Fest, July 11, 2020, Virtually hosted.



A Zoom window screenshot of a person standing in front of a greenscreen projection of the p5 Editor.

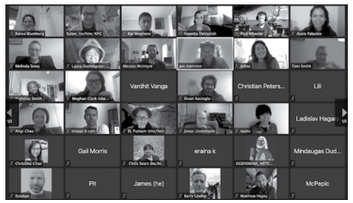


A Zoom window screenshot of a person standing in front of a greenscreen projection of a color scheme generator website.

Due to the onset of COVID-19, CC Fest went virtual. Audiences from all over the world gathered online and spent the day engaging in creative coding. Workshops, show-and-tell, and breakout rooms were facilitated. Keynote speakers and workshop leaders included Ashley Jane Lewis, Chris Orban, Aren Davey, Liam Baum, and Kofi Oduro. All donations from RSVPs went to TRANSIT ARTS, a youth arts development program.

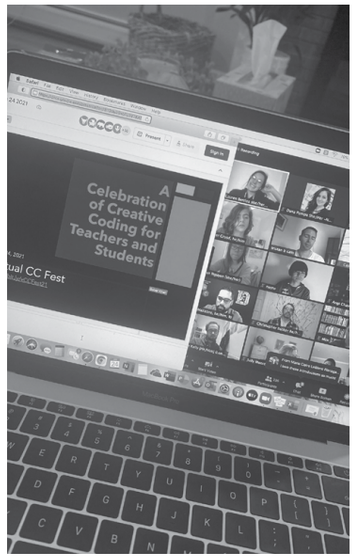
EDU 121 Creative Coding Fest, January 24, 2021, Virtually hosted.

#VirtualCCFest returned with twelve sessions on p5.js, Python, Hydra, Processing, ethics, AI, high school math, making games collaboratively, computational photography, and more. Keynote speakers and workshop leaders included Cassie Tarakajian, Jordan Harrod, Vivien Nguyen, Angi Chau, Liam Baum, Jeff Olson, Tristan Bunn, Flor de Fuego & Naoto Hieda, Nick McIntyre, Meghan Clark, Kofi Oduro, Katy Garnier, Joe Mazzone, and Derrick McMillen.



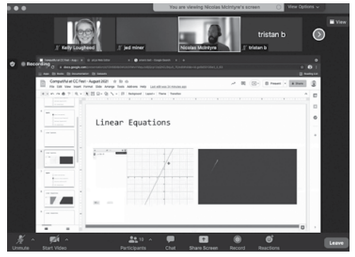
A screenshot of a Zoom window in grid view showing 36 tiles of attendees.


A laptop displaying a Google Slides presentation within a Zoom window. The current slide in the presentation is titled "p5.js Community Statement."

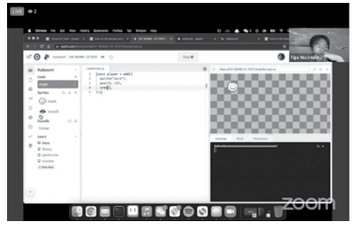



A laptop displaying a Google Slides presentation on the left of the screen and fifteen Zoom tiles of individuals on the right. The slide in the presentation reads "A Celebration of Creative Coding for Teachers and Students."

EDU 122 Creative Coding Fest, August 22, 2021, Virtually hosted.




A Zoom window displaying attendees and a presentation. The current slide reads "Linear Equations."



A Zoom window displaying a code editor.



Keynote speakers and workshop leaders included Miles Berry, Evelyn Masso, Dr. Emily Thomforde, Nick McIntyre, Tristan Bunn, Kofi Oduro, Ted Davis, Flor de Fuego & Naoto Hieda, Joe Mazzone, Angi Chau, and Derrick McMillen.

EDU 123 NYC DOE Curriculum, 2017, Luisa Pereira, with Teaching Contributors Courtney Morgan and Jose Orea.

Developed by the NYCDOE CS education team, the Introduction to Computational Media syllabus is a yearlong (108 hours) creative computing course for high schools using the open source JavaScript library p5.js. By understanding how code can be a medium for creative expression, students learn the fundamentals of computer science while designing and prototyping interactive projects that run on a browser. Additionally, students learn how HTML/CSS elements can interact with p5.js to fully take advantage of developing content for a browser. This course is currently being implemented in NYC public high schools via CS4All's Software Engineering Program (SEP), CS4All ICM Cohorts, and it aligns with the CS4All Blueprint for CS education.

EDU 124 createCanvas, October 15, 2019, Dan Shiffman.

createCanvas kicks off with an in-depth, two-part interview with Dan Shiffman! Dan is the beloved host of The Coding Train, the vibrant YouTube channel of weekly creative coding tutorials. Dan has been part of the Processing Foundation since before it was a foundation. In this episode, he talks to Education Community Director Saber Khan about how and why he started making educational materials for creative coding, what open source contribution can look like (spoiler: almost anything!), and takes us behind the scenes of his YouTube channel.

EDU 125 createCanvas, November 15, 2019, Dan Shiffman, Part 2.

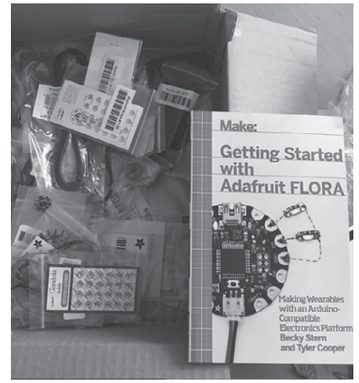
createCanvas returns with Part 2 of our in-depth interview with Dan Shiffman! In Part 2 of the interview, he talks to Education Community Director Saber Khan about different sustainability models for open source, the pros and cons of using YouTube as a platform, and how The Coding Train is more about community and documentation than it is about technical expertise.

EDU 126 createCanvas, December 13, 2019, Sharon Lee De La Cruz, Part 1.

Sharon is smiling and sitting on the floor in front of a bright painting featuring three Black girls.

Sharon Lee De La Cruz is a multidisciplinary artist and activist from New York City whose thought-provoking pieces address a range of issues related to tech, social justice, sexuality, and race. In Part 1 of her interview with createCanvas, Sharon talks about joyful resistance and what decolonizing freedom might look like. She discusses her focus on restructuring the dynamics of the classroom, to ensure that POC and other marginalized students don't have to rely on luck to have access to education.

EDU 127 createCanvas, January 16, 2020, Sharon Lee De La Cruz, Part 2.




A box filled with thread and sensors. A book lies on top of the box titled, "Getting Started with Adafruit FLORA." The subtitle in the bottom right reads "Making Wearables with an Arduino-Compatible Electronics Platform Becky Stern and Tyler Cooper."

In Part 2 of the interview with createCanvas, Sharon De La Cruz gets into the nitty gritty of what decolonization looks like in the classroom. She discusses how educators with privilege can approach working with marginalized communities, and how folks from those communities themselves are impacted by systemic racism in practice and ideology.

EDU 128 createCanvas, March 17, 2020, Aankit Patel, Part. 1.

Aankit sits at a large wooden desk between the American flag and New York State flag.

Aankit Patel talks about his previous role as Senior Director of Computer Science Academics at the NYC Department of Education, and how he and his colleagues have found that creative, physical, and critical computing is an approach that jibes with both teachers and students. Aankit also shares ways that teachers can approach their work with a sense of community practice.

EDU 129 createCanvas, April 17, 2020, Aankit Patel, Part 2.


Screenshot of a webinar that says "Introduction to Computational Media with p5.js." The text is green, yellow, and pink on black.

In Part 2 of his interview, Aankit Patel speaks about the different scales of impact that open-source software can have on education, as well as how to engage teachers in creating not only curriculum but a sense of community and agency for themselves.

EDU 130 createCanvas, May 20, 2020, Kelly Lougheed, Part 1.

Kelly is in front of a rainbow pixel background.

Kelly Lougheed talks about middle school and high school computer science education at all-girls' schools.

EDU 131 createCanvas, July 3, 2020, Kelly Lougheed, Part 2.

Kelly stands in front of the classroom looking at the camera. There are about six students around her.

Kelly Lougheed talks about creative applications of coding in the classroom and professional development and pedagogy for educators teaching coding.

EDU 132 createCanvas, October 1, 2020, Lauren Lee McCarthy.

In the first episode of Season 2 of createCanvas, Saber Khan talks with Lauren Lee McCarthy about p5.js, teaching, and being a student and a teacher.

Lauren's face is warped, with 2.5 orbs in front of her. The orbs show interiors of people in their homes.

EDU 133 createCanvas, October 30, 2020, Ari Melenciano.

Ari smiles at the camera. She is standing outside in the sun on a basketball court.

Ari Melenciano speaks about her experience teaching everything from high school AP computer science, kindergarten, and middle school to now computer science at the undergraduate and graduate university levels. Growing up an artist, she explains her approach to art-making within a context of education and activism, as well as a way to move through life. She discusses Afrotectopia's origins, and what she has learned about community-building and organizing over the years, as Afrotectopia has expanded to include a festival, fellowship program, summer camp, and, in January 2020, The School of Afrotectopia, a program that offered 10 free courses to over 250 students.

EDU 134 createCanvas, November 30, 2020, Sara Hendren.

Sara Hendren is an artist, design researcher, and writer who teaches design for disability. Sara discusses how to shift thinking and teaching around disability from assistive to adaptive design.



Headshot of Sara with brown hair smiling. She wears a jacket with a collar turned up.

EDU 135 createCanvas, February 5, 2021, Art Simon.

Art is standing and leaning down to look at the electrical project in his hands. A seated student is also looking at it.

In 2021, createCanvas turned into interview write-ups that continue to be published on Processing's Medium. In this interview with educator Art Simon, Art reflects on his nearly 30-year journey in computer science education. Art speaks on his experience in teaching Processing and p5.js to his high school computer science classes: "You can take relatively simple structures in the [Java] language and create engaging programs."

EDU 136 createCanvas, February 5,2021, Melanie Hoff.

Melanie Hoff is an artist and educator whose work recodes conventions of norms, interfaces, and sex, through software installation and new choreographies of exchange. In this interview, they talk through the community spaces they've cultivated, the workshops they've taught, and rethinking code as systems of communication, love, and care. "It's being able to see the ways that our lives are altered by codes that other people have written. When you understand how other people wrote them, then you can start to write your own for yourself and for the people that you care about around you."

A row of folks with laptops open. There is a group of three standing behind.

EDU 137 createCanvas, June 1, 2021, Tega Brain and Golan Levin.


A book cover, which includes the title and author names in white type, against an illustration of whimsical shapes of many colors on a dark blue background. The book is titled *Code as Creative Medium.*

Artists and educators Tega Brain and Golan Levin sit down with Saber Khan to discuss their book, *Code as a Creative Medium: A Handbook for Computational Art and Design.* Tega reflects on creating a textbook for the educator: "The institutions we work for don't provide that much support on how to teach these topics. We wanted to create something that would be useful, and [that] also speak to the fact that everybody has challenges, or similar stories, about what works and what doesn't. What happens on your worst day – how do you deal with that?"

Our mission is to promote software literacy within the visual arts, and visual literacy within technology-related fields — and to make these fields accessible to diverse communities. Our goal is to empower people of all interests and backgrounds to learn how to program and make creative work with code, especially those who might not otherwise have access to these tools and resources. The Processing Foundation was created in 2014 by Ben Fry, Casey Reas, and Daniel Shiffman.

COL 001

## Choreographic Coding Lab, 2015, Led by Motion Bank in Partnership with the Processing Foundation, Hosted by UCLA Design Media Arts and the UCLA Center for the Art of Performance.

A person performing a dance in front of a crowd.

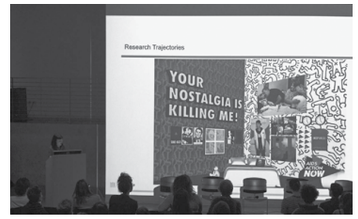
The Choreographic Coding Lab (CCL) was a workshop of exchange and collaboration for digital media artists working with code, who also have an interest in choreography, dance, and movement. CCLs are an outcome of Motion Bank, a four-year research project of the Forsythe Company focused on the creation of digital dance scores with guest choreographers.

A projection of someone standing in a pile of sand and waving a flag. A silhouette of an individual standing in front of the projection.

COL 002

## Biased Data: A Panel Discussion on Intersectionality and Internet Ethics, 2015, Organized by voidLab, Hosted by the Processing Foundation and UCLA Design Media Arts.

A person at the podium giving a presentation. The presentation slide consists of four portraits of people with Google Search autosuggestions covering their mouths.




A person at the podium presenting in front of a crowd. Their presentation is projected on the screen that reads, "Your Nostalgia is Killing Me!"

The Biased Data panel examined how real-world biases and inequality are replicated and systematically integrated into "neutral" algorithms and databases. Panelists included Safiya Noble, Marika Cifor, and An Xiao Mina. Casey Reas and Johanna Hedva co-moderated the panels.

COL 003

## Open Tech Lab, 2016, Organized by Emma Cunningham, Johanna Hedva, A.M. Darke, and the Feminist Center for Creative Work.

Open Tech Lab was a night of fun and informal events aimed to support communities of specific groups who have been historically marginalized in the fields of tech, coding, art, and online activism. Panel speakers included A.M. Darke, Ann Hirsch, Patricia Realini, and Seren Sensei.

A photo of five individuals dressed as cyberpunks projected onto a screen.

COL 004

## Scope Lab, 2017, Led by Miriam Posner and Lauren Lee McCarthy, with student research assistants Stalgia Grigg, Christina Yglesias, and Hillary Cleary.

Scope Lab was a workshop series hosted by UCLA Design Media Arts that focused on exploring code as a creative medium with which to understand and represent diverse perspectives. The studies were framed by the questions: "Whose perspectives are represented?", "Who has access to the tools to learn and

express themselves?" and "How do we design tools and projects that are more inclusive?" Each workshop consisted of hands-on programming experiments, lecture and discussion, and projects developed collaboratively. Prototyping took many forms including paper prototypes, software sketches with p5.js, and performances with physical objects.


Scope Lab zine designed by Hillary Cleary. A pile of Scope Lab zines splayed on a table.


The Scope Lab zine is opened to a spread of abstract pink and blue shapes. The right page is in focus with text on a wavy path that reads, "We ask questions like."

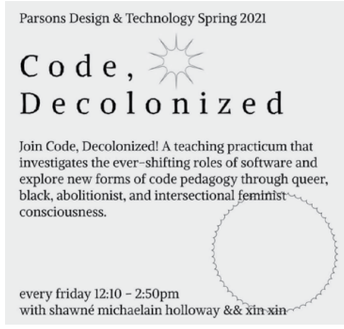

The Scope Lab zine is opened to a spread. The left page has text that reads, "Whose perspectives are represented?" and the right page has text on a circular path that reads, "How do we design tools that are more inclusive?"

A person sitting around a table with their laptop. The screen above them projects a slide of different weave patterns.


Students sitting around a table with laptops. A screen at the front of the classroom is projecting a slide that reads "UNCERTAINTY."

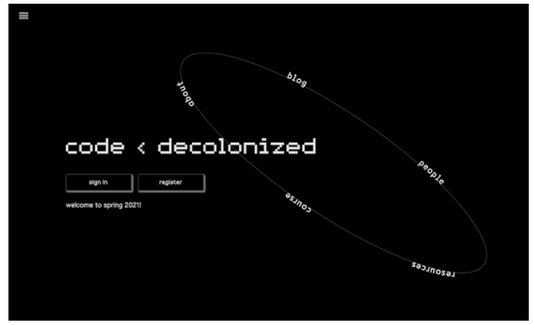
COL 005 Code, Decolonized, 2017, Taught by shawné michaelain holloway and Xin Xin, Mentored by A.M. Darke, Masood Kamandy-Milne, and Lauren Lee McCarthy.

Code, Decolonized is an initiative and a living archive that researches, experiments, and practices new ways of learning and teaching programming languages for the web through perspectives that are traditionally overlooked and underrecognized in the technical classroom. This collective work started from a teaching practicum course offered at Parsons Design & Technology at The New School, which investigates the ever-shifting roles of software and explores new forms of code pedagogy through queer, Black, abolitionist, and intersectional feminist consciousness. Following bell hooks's vision of "education as the practice of freedom," this course equips future educators with tools to reframe their understanding of traditional computer science education and craft socially engaged course materials.



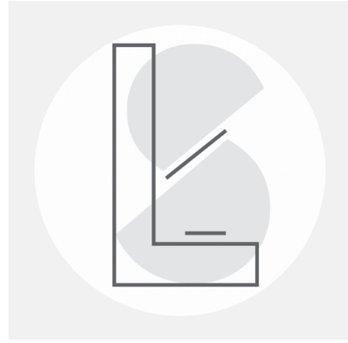

A yellow-green square with a title that reads "Code, Decolonized." The text underneath reads, "Join Code, Decolonized! A teaching practicum that investigates the ever-shifting roles of software and explores new forms of pedagogy through queer, black, abolitionist, intersectional, and feminist consciousness.






A screenshot of the webpage for the Code, Decolonized archive. White text on black background in loop shape.

COL 006 Logic School, 2021, Organized by Xiaowei R. Wang, Dorothy R. Santos, and Logic Magazine, In Partnership with Omidyar Network.



The logo for Logic School. The letter L is outlined in black and is overlayed on top of an abstract S comprised of two semicircles. The letters are encapsulated in a white circle.

Logic School is an online, experimental school for tech workers. Logic School's curriculum reflects the growing momentum around demanding change in the tech industry and draws from the worlds of activism, design, and software engineering. Designed in collaboration with the Processing Foundation, the program cultivates critical thinking about technology and its impact. It has a grassroots theory of change and believes the people who make the tech industry run—its workers—have the power to transform it.

When Ben Fry and Casey Reas started working on Processing in spring 2001, it was a personal project that had grown out of their work with John Maeda's Design By Number software. They wanted a new way of "sketching" code and a way to teach the fundamentals of coding to artists and designers. Ben released the first version in August of that year and the project has grown through bursts of energy since. Over the years, hundreds of people have contributed to Processing with their code and kindness. The pages that follow document different moments in the history of Processing and feature extraordinary contributions by members of the Processing community.

PRO 001 What is DBN?

1999.

Before Processing, there was Design By Numbers (DBN), a minimal and elegant coding system created by John Maeda. John invited Ben Fry and Casey REAS to work on DBN and their involvement inspired the first version of Processing in 2001. The following text is still served at https://dbn.media.mit.edu.

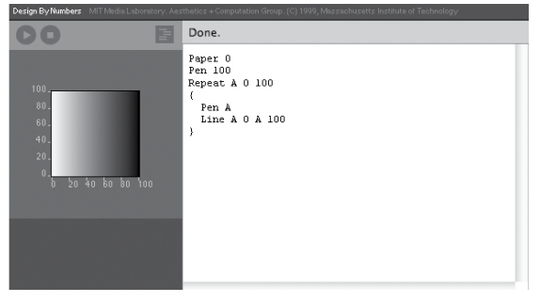



FIG 001-01 DBN editor, white-to-black gradient next to seven lines of code, paper pen line repeat.

Design By Numbers was created for visual designers and artists as an introduction to computational design. It is the result of a continuing endeavor by Professor John Maeda to teach the "idea" of computation to designers and artists. It is his belief that the quality of media art and design can only improve through establishing educational infrastructure in arts and technology schools which create strong, cross-disciplinary individuals.

DBN is both a programming environment and language. The environment provides a unified space for writing and running programs and the language introduces the basic ideas of computer programming within the context of drawing. Visual elements such as dot, line, and field are combined with the computational ideas of variables and conditional statements to generate images.

DBN is not a general purpose programming language like C or Java, but was designed to familiarize people with the basic concepts of computational media. Studying DBN is a first step to take–not a final step. Its advantages are:

- Free to use and multiplatform.
- Easy to understand syntax designed for beginners.
- Immediately accessible on the web.

There are three primary components to the Design By Numbers System:

- Design By Numbers core software: This software contains the complete DBN environment and can be viewed from the web or downloaded to an individual's computer.
- Design By Numbers book published by MIT Press Takes readers step by step through the DBN language with explanations of examples.
- Design By Numbers Courseware:
  A flexibly designed website generator built for educators who want to use DBN to teach computational design.

PRO 002

Proce55ing,
2001.

The earliest version of www.proce55ing.net (the original URL for the Processing website) was captured by the Internet Archive on 30 November 2001.

The homepage of the website featured the following short description.

Proce55ing is an environment for programming images, movement, and interaction. It is a sketchbook for developing ideas, a tool for creating prototypes, and a context for learning the fundamentals of computer programming. The Pr0cess1ng environment is in its early stages and will continue to develop.

Pr()ce55ing is written in Java and enables the creation of Java Applications and Applets within a carefully designed set of constraints. It uses a 2D/3D Java rendering API that is a cross between postscript-style imaging in 2D and 3D rendering with OpenGL.

Pr0cess1ng is created by Ben Fry and Casey Reas. For more information email Ben and Casey: ben[at]proce55ing.net, casey[at]proce55ing.net.



PRO 003

Processing Defined,
2003.

The following text was published on the Processing website (www.proce55ing.net) in 2003 as an early overview of the basics of the project.

Introduction

The Processing project introduces a new audience to computer programming and encourages an audience of hybrid artist/designer/programmers. It integrates a programming language, development environment, and teaching methodology into a unified structure for learning. Its goal is to introduce programming in the context of electronic art and to open electronic art concepts to a programming audience. Unlike other popular web programming environments such as Flash and Director, Processing is an extension of Java and supports many of the existing Java structures, but with a simplified syntax. The application runs locally and exports programs to Java applets, which may be viewed over the internet. It is not a commercial production tool, but is built specifically for learning and prototyping.

Concept

Graphical user interfaces became mainstream nearly twenty years ago, but programming fundamentals are still primarily taught through the command line interface. Classes proceed from outputting text to the screen, to GUI, to computer graphics (if at all). It is possible to teach programming in a way that moves graphics and concepts of interaction closer to the surface. Making exercises created during learning viewable over the web supports the creation of a global educational community and provides motivation for learning. A “view source” method of programming enables the community to learn from each other.

The concept of Processing is to create a text programming language specifically for making responsive images, rather than creating a visual programming language. The language enables sophisticated visual and responsive structures and has a balance between features and ease of use. Many computer graphics and interaction techniques can be discussed including vector/raster drawing, 2D/3D transformations, image processing, color models, events, network communication, information visualization, etc. Processing shifts the focus of programming away from technical details like threading and double-buffering and places emphasis on communication.

Programming Language/Environment

Processing is a Java environment which translates programs written in its own syntax into Java code and then compiles to executable Java Applet 1.1 byte code. It includes a custom 2D/3D engine inspired by PostScript and OpenGL. The software is free to use and the source code will be made public. It runs on Windows, Mac OS

X, Mac OS 9, and Linux and the software is currently in Alpha release. The Beta release is scheduled for Summer 2003. Processing Version 1.0 focuses on teaching basic concepts of interactive networked computer graphics.

Processing provides three different modes of programming—each one more structurally complex than the previous. In the most basic mode, programs are single line commands for drawing primitive shapes to the screen. In the most complex mode, Java code may be written within the environment. The intermediate mode allows for the creation of dynamic software in a hybrid procedural/object-oriented structure. It strives to achieve a balance between features and clarity, which encourages the experimentation process and reduces the learning curve.

Skills learned through Processing enable people to learn languages suitable for different contexts including web authoring (ActionScript), networking and communications (Java), microcontrollers (C), and computer graphics (OpenGL).

Networked Learning

The Processing website houses a set of extended examples and a complete reference for the language. Hundreds of students, educators, and practitioners across five continents are involved in using the software. An active online discussion board is a platform for discussing individual programs and future software additions to the project. The software has been used at diverse universities and institutions in cities including: Boston, New York, San Francisco, London, Paris, Oslo, Basel, Brussels, Berlin, Bogota, Ivrea (Italy), Manila, and Tokyo.



PRO 004 Processing Relase Notes #69,
2004, Ben Fry.

Each version of the Processing software comes with release notes written by Ben Fry. These early notes include insights into the goals of the original software side by side with detailed technical notes.

PROCESSING DEVELOPMENT ENVIRONMENT

(c) 2001-03 Ben Fry and Casey Reas Massachusetts Institute of Technology and Interaction Design Institute Ivrea.

RELEASE NOTES & DEVELOPER SOAPBOX

herein follows lots of random notes about the alpha releases of processing. more up-to-date details can be found in “revisions.txt” which has notes about individual releases.

‘revisions.txt’ contains more information about the specific updates and fixes in this release.

you’ll have to pardon the chatty detail in some spots, as this will also serve as a response to many of the ‘frequently asked questions’ that we have.

GETTING STARTED

double click the ‘Processing’ application, and select something from the examples menu: File -> Open -> Examples. hit the ‘run’ button (which looks like the play button on a vcr or tape deck). lather, rinse, repeat as necessary.

THANKS TO...

thanks to the many people who have been helping us out. it’s huge. i’ll get a nice long list of y’all in here soon.

REVISIONS & ROADMAPS

at least until the final “1.0” version, we’ll be using four digit numbers for the release. we are calling revision “0043” the first “alpha”, which for us means “first publicly consumable app that can be used by early adopters”. later revisions (like this one) will simply be numbered. the numbered releases aren’t heavily tested, so don’t be surprised if/when something breaks.. just report the problem and go back to the previous numbered release until there’s a fix.

there will be a few more numbered releases leading up to a beta release. beta means that all the features are in, but not all the bugs are out. there are several known issues with the alpha release (thin lines, lack of alpha transparency, etc) that will need to be sorted out for beta.

additional numbered releases will follow, leading up to 1.0, a version that we can actually proud of and that has a minimum number of bugs. hopefully this is not a *long* ways off, but...

I FOUND A BUG!

a cultured software elite such as yourself should use the gentleman's term "issue."

first, be sure to check under the notes for your specific platform to make sure it isn't a known issue or that there isn't a simple fix.

note! avoid the urge to just email us at processing@media.mit.edu, or sending mail to ben or casey directly. while you may prefer the privacy of an email, it is much quicker for you to ask the whole gang, who are super helpful. we also what we use to keep track of bugs, so we may just ask you to use the bboard anyway.

ok where was i.. next, check the bboard to see if something related has been re ported, or if there is already a workaround.

best method is to post to the bulletin board at: http://processing.org/discourse/ we prefer for you to use the bboard for bugs, since:

- we like to use the bboard as a way to track bugs and get feedback
- casey and ben can't always respond quickly to email
- and there are several knowledgeable people on the bboard

if you want to go straight to the bugs page, it's: http://processing.org/discourse/yabb/YaBB cgi?board=Proce55ing_software_bugs

when reporting this "bug" please include information about

1. the revision number (i.e. 0048)
2. what operating system you're using, on what kind of hardware
3. a copy of your code--the smallest possible piece of code that will produce the error you're having trouble with.
4. details of the error, which may be the last few lines from the files stdout.txt or stderr.txt from the 'lib' folder.

for stranger errors during compile time, you can also look inside the "build" folder inside "lib", which is an intermediate (translated into java) version of your code.

the more details you can post, the better, because it helps us figure out what's going on. useful things when reporting:

- we want the minimum amount of code that will still replicate the bug. the worst that can happen is we get a report that says "problem!" along with a three page program. sure, everyone likes a puzzle, but simpler code will be a faster response.
- occasionally we may need you to pack up a copy of your sketchbook or something similar so that we can try and replicate the weirdness on our own machine rest assured, we have no interest in messing with your fancy creations or stealing your ideas. the p5 team is a pair of straight-laced boys who hail from the midwestern u.s. who were brought up better than that. and as we often lack enough time to build our own projects, we have even less time to spend figuring out other peoples' projects to rip them off.



GOODIES & SEMI-HIDDEN FEATURES

- shift-click on the 'run' button to go straight to 'present' mode
- for quick renaming, just click on the sketch title
- inside the 'lib' folder is a 'pde. properties' file, which contains a handful of settings for your app and how it's set up. you can change the coloring of things, or even change your sketchbook location inside this file. a second file with a similar title but that includes "windows" or "macosx" etc in the name is for tweaks specific to your platform. for instance, we use the macosx-specific properties file to set the font size a little differently than on windows.

PLATFORMS

the processing development environment runs best on:

1. windows 2000/XP
2. mac os x
3. linux
4. mac os 9
5. windows 95/98/ME

our priority for how well the beast runs looks like:

1. windows & mac os x (tied for first)
2. mac os 9
3. windows 95/98/ME (because we must)
4. linux

windows is the superior platform for running java applications. it's not because we like windows the best, (sorry to the zealots in all other corners of the machine space) but that's just how it is. the vm for mac os x is really quite good (especially when compared to apple's previous efforts), but it's still a bit behind. we think os x will be a great bet for the future, and apple is putting all their feebleweight behind it, so hopefully it will evolve somewhere.

developing the version for mac os 9 is a big headache, but we think lots of people still use the crusty operating system, so we're going to keep supporting it for the meantime. the guess is that a lot of schools are still using it in their labs, and since schools are a significant target for the environment, we gotta play along. in the short term, however, development for mac os 9 has been suspended.

windows 95/98/ME is a piece of crap, but since lots of people (are often forced to) use it, we'll try and support. early alpha versions seem to be having trouble with 95/98/ME, but it'll run better in the future.

for the linux version, you guys can support yourselves. if you're enough of a hacker weenie to get a linux box setup, you oughta know what's going on. for lack of time,

we won't be testing extensively under linux, but would be really happy to hear about any bugs or issues you might run into. actually, we don't get happy that you're having issues, but if you're going to have issues, we're happy that you tell us about them, so we can fix them.

MAC OS X

the most current release has only been tested on Mac OS X 10.2.6. your mileage may vary if you're running something else. actually, your mileage will vary no matter what, because who knows what this software is gonna do. you're playing with free, alpha software. get psyched!

minimum requirements.. processing requires at least Mac OS X 10.1. if you're running anything older than 10.2, you'll need "Java 1.3.1 Update 1", the latter of which is available as a free update from the "Software Update" control panel. it can also be downloaded from http://www.apple.com/downloads/macosx/apple/ or from: http://www.apple.com/downloads/macosx/apple/java131.html for what it's worth, we don't test processing under mac os x 10.1 and we don't recommend it at all.

mouse wheel support only works if you're using java 1.4. the latest version of java will be available via the software update control panel.

(actually this paragraph is only relevant if you want to try java 1.4, since we wound up using 1.3 as the default for release 58) if you're having random troubles (exceptions being thrown, screen painting weirdness, general confusion) you might want to try running processing with java 1.3.1 instead of java 1.4. to do so, right-click or control-click the processing application and select "Show Package Contents". go to Contents -> Resources -> and then open MRJApp.properties in a text editor. remove the # from this line: com.apple.mrj.application.JVMVersion=1.3.1 and add a # in front of this line: com.apple.mrj.application.JVMVersion=1.3+

serial port.. we use rxtx (version 2.1_6) to handle serial i/o, which is included with the processing release. unlike previous releases (anything before 57), it no longer requires separate installation. however, if this is the first time you're using rxtx, you'll need to run serial_setup.command (double-click it and follow the instructions) to make sure that things are properly set up (a few permissions need to be changed). if you're getting a "serial port is already in use by another application" it's possible that you haven't run this script. you may also need to reboot after running the script. on my machine, i installed the keyspan driver for my usb-serial converter, ran the script, and then rebooted in order for things to work. in the past, i've used a keyspan 28X dual port adapter, and the selection i use on the serial port menu reads "/dev/cu.USA-28X21P1.1." you'll probably have something similar. don't mind the frightening names. another note on serial port.. tom igoe was kind enough to note that you'll be in a world of



hurt if you disconnect your serial adapter while a sketch is running--it'll prolly freeze the machine and require a forced reboot. (while this may seem nutty, you might run into it if your adapter is plugged into your usb keyboard, and you have the keyboard plugged into a monitor/keyboard switcher).

quitting presentation mode.. on other platforms, hitting the 'escape' key will quickly get you out of presentation mode. however, there seems to be some key event weirdness under osx. we hope to find a fix someday.

MAC OS 9

we have temporarily suspended development for mac os 9, because we don't have time to fight with this dying os before beta. we hope to resume mac os 9 development before releasing the final 1.0 version.

for releases earlier than 57:

java applications on classic mac os are in a bad state, as apple has decided (rightfully so) to abandon further development of their java runtime under OS 9.

serial works fairly well with my keyspan usb/serial adapter. thank god for patrick beard and jdirect.

versions: we only test under Mac OS 9.2.2, all others.. who knows?

WINDOWS

win2k and winxp are used as the primary development platforms, so the release will likely work best on either platform.

win95/98/me seem to have some trouble, but we think it's just with the .exe that we use, so that'll get fixed in the future. you can try using the 'run.bat' file instead, and see if that works better.

the release is now split into 'standard' and 'expert' versions. the basic release includes a working java vm, and is all set up and ready to go. the advanced version is for people who already have java installed (and don't want to deal with the 20MB download), and know what they're doing enough that they can also install the serial port code by hand. instructions on installing the serial code are in the 'serial' folder inside the 'expert' release.

out of memory? try adjusting the parameters in the file 'run.bat' and use that to run instead of Processing.exe. short instructions can be found inside that file.

mouse issues: by default, windows seems to skip every other pixel on screen, causing weirdness for some drawing applications done with p5. if you're seeing this, you can fix it by going to the windows "mouse" control panel, the "pointer options" tab, and select "enhance pointer precision." (this was actually tracked down by someone else in the p5 community, whose name i have misplaced. if it was you, please drop me a line so you can be properly cited. this kind of help is huge for us, since we're

such a small group!)

video.. if you want to use video input on the pc, you'll have to have a device that's compatible with quicktime, or use something like winvdig to make it work: http://www.vdig.com/WinVDIG/index.html otherwise you'll get no video.

"hs_err_pid10XX.txt" error.. this is something within the java vm that we can't fix. it's not clear what the problem is, but it seems to have shown up with java 1.4.

LINUX

the processing application is just a shell script, you can use this as a guide to getting p5 to run with your specific configuration, because who knows what sort of setup you have. this release was tested on a redhat 9 box, and sun's jre 1.4.2 is included with the download. replacing (or making a symlink to) the contents of the 'java' folder will let you tie in a preferred jvm for your machine.

jikes.. just as 58 was being released, we ran into a problem where jikes (the java compiler used by p5) couldn't be found by the application on linux. faced with the deadline, we decided to put up an error message saying it wasn't found. you should make sure jikes version 1.18 (we strongly recommend this specific version!) is installed on your machine and in your path.

serial.. this release uses rxtx-2.1_6 (just like macosx). you may get error message spew to the console when starting the application saying "Permission denied" and "No permission to create lock file" and to read "INSTALL". this is because you need to add yourself to either the uucp or lock group so that processing can write to /var/lock so it doesn't get in a fight with other applications talking on the serial port. supposedly, adding yourself to one of these groups will work (didn't for me, but i'm a little clueless) or running processing as root will get rid of the errors (not a great solution).

WHAT IS SKETCHBOOK?

we think most "integrated development environments" (microsoft visual studio, codewarrior, jbuilder) tend to be overkill for the type of audience we're targeting with Processing. for this reason, we've introduced the 'sketchbook' which is a more lightweight way to organize projects. as trained designers, we'd like the process of coding to be a lot more like sketching. the sketchbook and the 'history' menu under 'sketch', are attempts in that direction.

WHY JAVA? OR WHY SUCH A JAVA-ESQUE LANGUAGE?

We didn't set out to make the ultimate language for visual programming, we set out to make something that was:

1. a sketchbook for our own work, simplifying the majority of tasks that we undertake,
2. a teaching environment for that kind of process, and



3. a point of transition to more complicated or difficult languages like full-blown Java or C++. (a gateway drug)

At the intersection of these points is a tradeoff between speed and simplicity of use. i.e. if we didnt' care about speed, python or other scripting languages would make far more sense. if we didn't care about transition to more advanced languages, we'd get rid of the crummy c-style (well, algol, really) syntax. etc etc.

Processing is not intended as the ultimate environment/language (in fact, the language is just Java, but with another graphics api and some simplifications), it's just putting together several years of experience in building things, and trying to simplify the parts that should be easier.

EXTERNAL FILES / FONTS / READING DATA FILES

if you want to use external files, like images or text files or fonts, they should be placed in a folder called 'data' inside: sketchbook -> default -> SKETCH_NAME

starting with version 44, there are several functions that make dealing with data in files much easier (loadFile, loadStrings, splitStrings, etc) so file i/o should be fun!

WHY IS IT CALLED "PROCESSING"?

at their core, computers are processing machines. they modify, move, and combine symbols at a low level to construct higher level representations. Processing allows people to control these actions and representations through writing their own programs.

the project also focuses on the "process" of creation rather than end results. the design of the software supports and encourages sketching and the website presents fragments of projects and exposes the concepts behind finished software.

"Proce55ing" is the spelling we formerly used for the url (processing.net being unavailable) and while it's a combination of numbers and letters but is simply pronounced "processing." you also might see "p5" used as a shortened version of the name.

PROCESSING IS FREE TO DOWNLOAD / FREE TO USE

we think it's important to have Processing freely available, rather than selling it for a million dollars under some godawful yearly contract update scheme. to that end, we encourage people to distribute the word widely and refer them to the site: http://processing.org

on most of our own projects, we usually list them as "Built with Processing" or something similar, with a link back to the site. of course this isn't a necessity, but it makes us happy when you do.

SOURCE CODE / OPEN SOURCE / GPL BLAH BLAH

we plan for this project to be "open source", everyone's favorite phrase that means that you'll be able to get your grubby little mitts all over our code (all the code that's behind

the processing development environment and the graphics engine used in tandem with it). we can't promise, since we're still working on getting the licensing taken care of with our employers, but we think this should likely happen soon.

the export libraries (internally known as 'bagel') will probably be LGPL, which means they can be used as a library and included in your project without you having to open up your code (though we encourage people to share anyway).

more information about the gnu public license can be found here: http://www.gnu.org/copyleft/gpl.html

processing also includes other open projects, namely the oro matcher, the kjc compiler, and the jedit syntax package. the oro tools are distributed under a bsd style license as part of the apache jakarta project, and the kjc compiler is part of the kopi suite of tools, which is released under the gpl. so in fact, if the final, publicly available version of processing still uses kjc, the code for processing will have to be released gpl. more about the oro tools is at: http://www.savarese.org/oro/ and the home for kopi/kjc is here: http://www.dms.at/kopi/

kjc is being phased out in favor of the jikes compiler from ibm: http://oss.software.ibm.com/developerworks/opensource/jikes/ which is covered by the ibm public license.

we're sorry that our source code isn't available just yet, we're cleaning and scrubbing it, it was a decision between getting the alpha out to people to try versus taking a few more weeks to clean up the project and deal with the technology licensing departments at mit and ivrea. these things are far more difficult and time consuming than they would appear.

our plan is to have the code available with the first "beta" release, which will be the first release that is publicly available and downloadable from the site.



PRO 005 Processing vs. Lingo Comparison, 2003.

When Processing was new, we created comparison tables between Processing and other coding languages. We had comparisons for Java, ActionScript, Lingo, Python, and Design By Numbers on the website. These helped people to convert what they already knew about coding to Processing and to bring their knowledge of Processing to other environments.

Lingo is the language written for Macromedia's Director software. Director was the dominant environment for designers and artists making CD-ROM projects, but has declined in popularity during the web era due to the success of Flash. It is still one of the most commonly used environments and it has excellent libraries of code for extending its functionality.

Lingo is integrated into a visual environment with a theater metaphor using terms like "stage" and "cast" to describe different elements of the software. The Lingo language is characterized by its verbose English-like syntax. It has been modified in recent years to support object oriented structures and 3D graphics.

Online here: https://web.archive.org/web/20030605073321/http://proce55ing.net/reference/compare/lingo.html

PRO 006 Early Processing Support, 2001–2012.

Prior to incorporating as a foundation, Processing received key funding and support from several organizations and companies. These commitments enabled a series of pivotal improvements to the software.

FIG 006–01 From left to right, Casey REAS, Ben Fry, Dan Shiffman at a bed and breakfast in Indiana while working on Processing 1.0 in nearby Oxford, Ohio.

Three people sit at a table with flowers and 3D words “faith,” “love,” and “believe” in front of them.

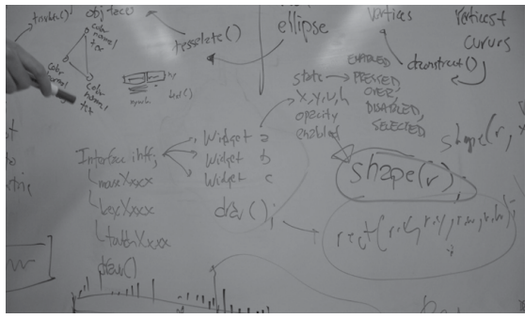



FIG 006–02 Notes while working on Processing 2.0 at NYU in 2011.

Marker on whiteboard, many arrows, words shape, ellipse, curves.

The Interactive Telecommunications Program (ITP) at New York University sponsored a Processing 2.0 development workshop in the summer of 2011.

The Armstrong Institute for Interactive Media Studies at Miami University funded the Oxford Project, a series of Processing development workshops during the 2008–2009 academic year. These four-day meetings in Oxford, Ohio, enabled the November 2008 launch of Processing 1.0 and stimulated future development.

Oblong Industries funded Ben Fry to develop Processing during the summer of 2008. This funding also assisted in the completion of the 1.0 release. The Rockefeller Foundation awarded Ben Fry with a Media Arts Fellowship in 2006. The grant marked the first time that Ben was able to work on Processing as a funded project. This support led to further developments of the OpenGL and PDF rendering engines, as well as significant enhancements to other libraries and their integration.

The Interaction Design Institute Ivrea funded four individuals’ work on Processing in the summer of 2003. This resulted in Dan Mosedale’s preprocessor using Antlr, Sami Arola’s contributions to the graphics engine, and other contributions to the Processing Development Environment and 2D graphics engine. We are grateful to Interaction Ivrea director Gillian Crampton Smith for her encouragement and support.



PRO 007 Processing 3.0 in Denver, 2014, Chris Coleman.

FIG 007–01 From top to bottom, Dan Shiffman, Manindra Moharana, Ben Fry, and Andres Colubri. Four men sit on stairs with laptops.

While it presents as a simple and easy-to-use-tool, development of Processing is a complex combination of code writing, documentation updates, multi-platform builds, and web deployments. As with any open-source project, coordinating all of these elements for a major release across multiple time zones and work schedules is an arduous task. This is why emphasis remains on bringing people together in person when shared timelines, singular focus, and quick communication enables leaps forward for a project. In late 2014, five members of the Processing development team gathered in Denver, Colorado to do just this. In planning the release of Processing version 3, I offered to host Processing at the facilities of the Emergent Digital Practices (EDP) Program where I am a professor. We at EDP gathered funds from the Program to support the open source tools we use in our classrooms, specifically Processing, Processing for Android, and p5.js. This enabled Andres Colubri, Ben Fry, Manindra Moharana, Casey Reas, and Dan Shiffman to fly to Denver and devote five days to the release of v3. The team gathered in our small graduate seminar classroom, scrawled notes across multiple whiteboards, and worked morning to night. They took brief breaks for coffee and meals provided onsite from local Denver restaurants and organized by Professor Laleh Mehran and me. Dan Shiffman and Casey Reas also gave brief presentations and answered questions from EDP students about Processing and creative coding. While not all the people who worked on the release could be there in person, it was a testament to what can get done on open-source projects when we connect IRL. This was also the first major effort at the University of Denver to support open-source tools for the arts and inspired me to keep going, creating the Clinic for Open Source Arts in 2018.

Documentation from the Processing 3.0 development meeting sponsored by the Emergent Digital Practices program at the University of Denver, November 2014.

PRO 008 Processing 3.0 Release, 2015.

The Processing 3.0 launch in 2015 was a huge effort by an extended core team. The text printed here includes an email prepared for the launch, incomplete presentation notes on Casey's computer, and the ever-important list of "changes" to the new release.

FIG 008-01 Processing logo.

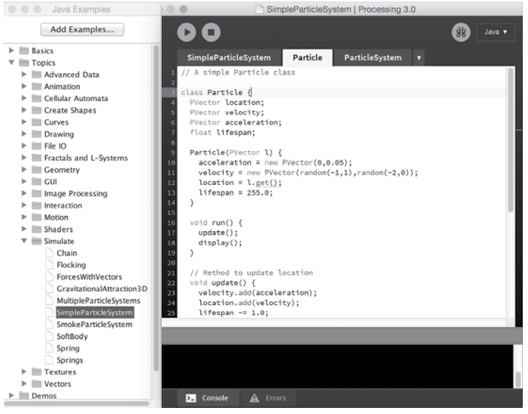



FIG 008-02 Processing editor with libraries menu open.

We're thrilled to invite you to try Processing 3. As always, it's free and open source and it's available here: https://processing.org/download/.

Processing 3 has a new code editor with an enhanced interface, better error checking, and a built-in debugger. Dan made a video to explain the debugger: https://vimeo.com/140134398. Processing 3 is faster for OpenGL (P2D and P3D) work and we've added new features for high-resolution screens and full-screen display. We also have a new and easier way to install and update Processing Libraries, Modes, and Tools.

The Processing 3 software was developed by the non-profit Processing Foundation. We started the Foundation <http://foundation.processing.org/> in 2012 with the two-fold mission to promote software literacy within the visual arts, and visual literacy within technology-related fields. Our primary goal is to empower people to learn how to program.

Individual contributions from the community, Google's Summer of Code, and Fathom Information Design are our greatest sources of support. We thank O'Reilly Media, the Interactive Telecommunications Program at New York University, and the Emergent Digital Practices program at the University of Denver for additional support. If you can support our work, please get in touch at foundation@processing.org.



Yours, Ben, Casey, Dan, Andrés, Scott, Elie, Florian, Jakub, and James

//

Casey Notes

1. Hello and welcome, my name is . . . .
2. What's new in Processing 3?
3. A brand new editor!
    1. new interface and look
    2. live error-checking, multiple error messages, better error messages
    3. auto-completion
4. OpenGL renderers rebuilt for speed and efficiency, no more Applet!
    1. fullScreen() with screens and SPAN
    2. I'd figure out a demo or two that shows off how speedy and flicker-free things are here... Andrés might have an idea for a few, i.e. a full-screen shader example and some others [fry]
5. Support for high density displays
    1. pixelDensity()
    2. JFX for retina and 2D work (experimental!)
6. size() has to use hard coded numbers, you can use settings() otherwise
    1. too technical? I hope people *never* have to use settings(), and am a little confused by the amount of confusion around this [fry]
7. new contributions manager—how to get libraries, video, sound as examples?
    1. combine libraries, tools, examples, modes
8. sketchbook migration, 2.0 and 3.0 sketchbooks
9. to do
    1. libraries not ported for 3.0, we've reached out and are hopeful
    2. this can also be a call for library/tool/mode authors—if they have questions they should get in touch b/c we want to support them [fry]
    3. mention example packages and get another plug in for shiffman books! [fry]

//

Changes in Processing 3.0

This page lists the most important changes between Processing 3 for people who are familiar with Processing 2. For more specific details, read the revisions.txt (https://github.com/processing/processing/blob/master/build/shared/revisions.txt) file to see what's changed since the release before it. This is where you go if your code stops working in the most recent release, or if you want to know if your favorite bug has been fixed.

The Big Stuff

Rendering rebuilt – OpenGL (P2D and P3D) is now stutter-free and very speedy. Some JAVA2D performance improvements as well. The new FX2D renderer offers huge speedups for 2D drawing, especially with high density "retina" displays.

New editor – The main editor window now includes:

Autocomplete! (can be activated in Preferences)

A full-featured, easy to use debugger

Tweak Mode (http://galsasson.com/tweakmode/) has been incorporated

New interface – It's not yet finished (https://github.com/processing/processing/issues/3072) but the UI is undergoing a major makeover.

High-res display support – New methods pixelDensity() anddisplayDensity() make it easier to create sketches that will run nicely on high-res ("Retina") displays. It sounds so simple when put like that, but this is a really big deal.

Unified Contributions Manager – We used to have separate windows for installing Libraries, Modes, and Tools. Now a single "Contributions Manager" helps you manage installation and updates for all of these contributions by third-party authors, plus... Examples!

Sketchbook Migration – If you already have a (2.x) sketchbook, 3.0 will ask if you want to create a new, 3.0-specific sketchbook, or share the existing one.

Things That May Break Your 2.x Sketches

Do not use variables in size() – This time we really mean it. We've been saying that size() must be the first line in setup since at least 2004, and that using variables instead of numbers will cause problems since 2006, and by at least 2009, simply "don't do it," but now the chickens have come home to roost. In the past, the size() function was implemented by doing backflips behind the scenes. Those backflips made things very fragile, introduced cross-platform quirks that have consumed too many of my weekends, and prevented us from making wholesale performance improvements to the rendering system (higher performance, better full screen support, etc). But despair not! If you must change the size of your sketch, use surface.setSize(w, h), which is the one and only (safe) way to alter your sketch's size. A short demo that's both resizable and gives you a random sketch window size whenever you hit a key:

```
void setup() {
  size(400, 400);
  surface.set
  Resizable(true);
}
```



```
void draw() {
  background(255);
  line(100,
  100,width-100,
  height-100);
}

void keyPressed() {
  surface.setSize
  (round(random
  (200,500)),
    round(random(200, 500)));
}
```

Applet is gone – Java's java.awt.Applet is no longer the base class used byPApplet, so any sketches that make use of Applet-specific methods (or assume that a PApplet is a Java AWT Component object) will need to be rewritten.

You only smooth once – smooth() and noSmooth() can only be used insetup(), and only once per sketch. Note that smooth() has enabled by default since 2.x, so it's unlikely you'll need it anyway.

For the curious or insomniac, this document (https://github.com/processing/processing/blob/master/core/README.md) has the technical details about why these changes were made.

New

Use the FX2D renderer for greatly improved 2D graphics performance. It has many improvements over the default renderer, though it has a few rough edges so we haven't made it the default. fullScreen() method makes it much easier to run sketches in, well, full-screen mode. The PVector class now supports chaining methods.

SVG Export that works just like the PDF Export

A new settings() method that is called behind the scenes. Most users will never notice this, but if you're using Processing without its preprocessor (i.e. from Eclipse or a similar development environment), then put any calls to size(), fullScreen(), smooth(), noSmooth(), and pixelDensity() into that method. More information can be found in the reference. Only users who are in other development environments should use settings(). It shouldn't be used for any other purpose.

Updated application icons.

Changed

The Video and Sound libraries are no longer included in the download (because they've

grown too large) and must be installed separately. Use Sketch → Import Library → Add Library... to install either one.

Deprecated

The variables displayWidth and displayHeight are now deprecated, despite being available in 3.0.

Removed

Lots of bugs

Known Issues and What's Coming

There are plenty of issues and we could use some help!

On Windows, launch4j doesn't work from folders with non-native charsets. On an English version of a Windows system, any characters in CP1252 are fine.

P2D and P3D windows behave strangely on OS X when larger than the screen size.

When using cursor() in P2D and P3D, the cursor images do not match what you expect from the OS.

When using Java 8u60 on Linux and OS X, smoothing is disabled for the JavaFX renderer. If you use the version of Java that's included with Processing, you'll be fine.

When using selectInput(), selectOutput(), and selectFolder() with OpenGL on Windows, the sketch window will close until the file is selected. We're waiting for an upstream fix from the JOGL project.

Changes for Libraries, Modes, and Tools

Some Libraries, and all Modes and Tools will need to be updated to be compatible with 3.0. We'll be expanding this documentation to cover those changes and doing our best to support authors of these contributions through the changes.

For the vast majority of authors, the changes are quite simple, and involve class name or package changes inside processing.app.

The exception is any Library where 1) assumptions were made about PApplet-being a subclass of Applet (and Component), or 2) relied on AWT-specific features in processing.core. More about the changes to core can be found here.

Modes and Tools need slight modifications because much of the UI code for Processing has moved from the processing.app package (which had grown unmanageably large) into processing.app.ui. This doesn't give us perfect separation of UI and non-UI code, but it's a helpful step in the right direction.

Several of the (static) utility functions from Base have moved into classes calledUtil, Messages, and Platform because Base was getting enormous. (It still is enormous, but it's now a wee bit more reasonable.) Since enough other things are breaking in 3.0, we're not including accessors for the deprecated version of the



functions, just making a clean break. Common changes will include: Base.isMacOS() or Base.isWindows() becomes Platform.isMacOS() andPlatform.isWindows() (sensible, right?)

Base.showWarning() becomes Messages.showWarning()

Base.log() becomes Messages.log()

A good rule of thumb is that if there are platform-specific qualities to it, it's probably in Platform. If it's a message (whether a dialog box or a log file), it's probably in Messages. And all those file utilities are in that Util class.

Modes (and some Tools) will also need to be updated based on the major UI changes in 3.0. The default "Java" Mode is now completely separate from the rest of the code, so it's a decent model for understanding how to use the newEditorButton classes and similar. If your Mode subclasses JavaMode, you'll want to check out how Android Mode does this and how it imports the necessary classes from the Java Mode, since they're no longer on the default CLASSPATH. Starting in 3.0 beta 6, each Tool is only initialized once. This saves lots of time and memory. In beta 7, the init() method was changed to pass a Base object instead of Editor, because it's necessary to call Base.getActiveEditor() whenever a Tool's run() method is called. See the Tool Basics page for an example.

The 2.0 and 3.0 lists of Libraries, Modes, and Tools are stored separately, so it's possible to maintain both versions, if you'd like to do so. Users also have the ability to use separate sketchbooks for 2.x and 3.x versions of Processing, so they can have separate versions installed while the transition happens.

Library authors, now is the time to reduce your reliance on AWT! In 3.0, we're moving away from AWT in a big way (here's why). Any library features that require AWT should be treated with suspicion. Modes and Tools can still use AWT, but the OpenGL renderers (P2D and P3D) and the upcoming FX2D renderer don't use AWT at all.

This is one reason we built the processing.event classes in 2.x, and have been removing spurious AWT usage from the core API, documentation, and examples. It's now been a couple years since we made those changes.

The other reason is that we can't rely on AWT features when targeting JavaScript or Android, so it was encouraging bad habits.

PRO 009 Processing Libraries, 2021.

Contributed libraries are created by members of the Processing community. They are independently and generously created to share. They are all open-source and include their own reference and examples. This is the list of Libraries pulled from www.processing.org on 1 October 2021.

Utilities

- $1 Unistroke Recognizer, Darius Morawiec. Implementation of the $1 Gesture Recognizer, a two-dimensional template based gesture recognition.
- Andrew's Utilities (AULib), Andrew Glassner, Motion blur, fields, easing, waves, uniformly spaced curves, globs, and more!
- ColorBlindness, Jim Schmitz. Fast and easy-to-use utility for simulating color blindness or performing daltonization.
- ColorScheme, J. Taylor OConnor. A tool to import color schemes.
- CountdownTimer, Dong Hyun Choi. A countdown timer which triggers callback events for each user-defined tick interval during the timer's duration.
- Dawesome Toolkit, Brendan Dawes. Convenient utilities for some everyday tasks including simple layout using grids, lat lon distance calculation, creating patterns with a Vogel spiral, color pallete generation, displaying guides, simple debug panel as well as useful PVector list manipulation.
- Free Transform, Bartosh Polonski. Transforms textures interactively.
- GR Infinidecimal Canvas, C. Sina Cetin. A numeric canvas that stores the image as a number array to create visuals with arbitrary ranges.
- ID3, Jorge C. S. Cardoso. Extracts ID3 tags from Mp3 files.
- Interfascia, Brendan Berg. Builds simple yet gorgeous user interfaces.
- Keystone, David Bouchard. A library to help with basic projection mapping.
- Live Brush, Boy d'Hont. A live-coding library that draws code brushes from external Java source code.
- Most Pixels Ever, Daniel Shiffman. Framework for spanning Processing sketches across multiple screens.
- Mother, Ilias Bergstrom. A library for live visuals performance with Processing sketches.
- Mouse 2D Transformations, Alex Poupaki. Calculation of mouse coordinates in transformed 2D animation matrices.
- Nest, Eric Socolofsky. Scenegraph and mouse event handling system based loosely on ActionScript 3.0.
- Nice Color Palettes, Federico Pepe. Use the best color palettes from ColourLovers.com in Processing.



- nub, Jean Pierre Charalambos. Library that eases the creation of interactive scenes.
- OOCSI for Processing, Mathias Funk. Processing client library for the OOCSI design middleware and Data Foundry access.
- P8gGraphicsSVG, Philippe Lhoste. Allows Processing to export the drawings of a sketch in SVG format.
- PortMods, Pike. Built for Processing 3, PortMods are pieces of Java code with input- and output ports that can be chained.
- PostFX for Processing, Florian Bruggisser. Framework for applying post effects to PGraphic objects.
- RunwayML Processing Library, George Profenza. Easily send and receive data between RunwayML and your sketches.
- Scratch, Mike Barkmin. Provides classes to match the functionality of Scratch.
- SketchMapper, J. Taylor OConnor. A GUI tool to map sketches onto surfaces.
- spacebrewP5, Brett Renfer. A toolkit for prototyping interactive spaces.
- TimedEvents, Jason Gessner. A couple of classes for firing off timed events at regular or random intervals.
- Timing Utilities, Lord of Galax. Library containing a number of useful classes for time-keeping in Processing.

Data

- (Weka4P) Weka Machine Learning for Processing, Rong-Hao Liang. Implementation of the Open Source Weka Machine Learning Java library for Processing 3.
- AndroidCapture for Processing, Jianbin Qi. This lib tries to transfer data (Android Camera & Android Sensor) between Processing and Android.
- ArtNet for Java and Processing, Florian Bruggisser. Framework for sending and receiving DMX data over the ArtNet protocol.
- BezierSQLib, Florian Jenett. A library to facilitate communication with SQL-based databases.
- Carnivore, RSG. Surveillance tool for data networks.
- Deep Vision, Florian Bruggisser. Deep computer-vision algorithms for Processing.
- GML4U, Jerome Saint-Clair. A Graffiti Markup Language library for Processing.
- HTTP Requests for Processing, Rune Madsen. A small library that takes the pain out of doing HTTP requests in Processing.
- HiVis, Philip Poronnik, Oliver Bown, Oliver Coleman, Phillip Gough, Narayan Sankaran. Import data and manipulate it with statistical and other functions. Designed to be simple to use for novice programmers while being highly flexible and extensible. Includes many examples and tutorials. Check out olivercoleman.github.io/hivis for a 30-second crash course.

- MapThing, Jon Reades. A collection of classes for reading and displaying Shape files (a.k.a. ESRI shapefiles), CSV, and GPX data in a map-based sketch.
- OOCSI for Processing, Mathias Funk. Processing client library for the OOCSI design middleware and Data Foundry access.
- QRCode, Daniel Shiffman. Reads QR Code images, a two-dimensional barcode format.
- Redis, Darius Morawiec. Wrapper to use Redis in Processing. It's based on Jedis, a small Java client by Jonathan Leibiusky.
- RunwayML Processing Library, George Profenza. Easily send and receive data between RunwayML and your sketches.
- SFTP, Daniel Shiffman. SFTP direct from Processing (using JSch).
- SQuelized, Samuel Alarco Cantos. A library to facilitate the connection between Processing and SQL databases.
- ShortMessage, Hamzeen. H. A library to send/receive short messages (SMS) from Processing.
- Signal Filter, Raphael de Courville. Filters noisy and jittery signals.
- Simple Tweet, Gottfried Haider. Posts images from Processing to Twitter.
- SoundCloud, Darius Morawiec. Unofficial Java library, which simplifies the use of the official SoundCloud Java API wrapper.
- Squarify, Agathe Lenclen. A squarify treemap layout generator.
- Temboo, Temboo. Generate code to connect to 100+ APIs, code utilities, and databases in Processing.
- UDP, Stephane Cousot. Enables simple UDP communication, as well as multicast support.
- Unfolding Maps, Till Nagel. Creates interactive maps and geovisualizations.
- WootingKeyboard, Shinhoo Park @ KAIST Interactive Media Lab. Analyzes the raw data of the Wooting keyboard.
- XlsReader, Florian Jenett. A library to read from XLS (Excel) files.
- oscP5, Andreas Schlegel. An Open Sound Control (OSC) implementation.

Simulation

- AI for 2D Games, Peter Lager. An AI framework suitable for 2D games and simulations.
- Box2D for Processing, Daniel Shiffman. A library and set of examples for 2D physics simulation wrapping some aspects of JBox2D, a Java implementation of Box2D.
- Carnivore, RSG. Surveillance tool for data networks.
- Combinatorics, Florian Jenett. Generate combinations, variations and permutations.
- Culebra Behavior Library for Processing, Luis Quinones. A collection of objects and behaviors for creating dynamic multi agent interactions.



- Fisica, Ricard Marxer. A wrapper for JBox2D, a 2D physics engine.
- LiquidFunProcessing, Thomas Diewald. RigidBody/Particle simulation using JBox2d/LiquidFun, a 2D physics engine.
- PixelFlow, Thomas Diewald. PixelFlow is a Processing library for high-performance GPU-Computing (GLSL), like Fluid Simulation, SoftBody Dynamics, Rendering, Optical Flow, Image processing.
- teilchen, Dennis P Paul. Simple physics library based on particles, forces, constraints and behaviors.

I/O

- AP-Sync, Nigel Tiany. Offers an easy way to sync data to processing from your Arduino or any microcontroller to processing and back.
- Augmenta for Processing, Theoriz. Allows you to use tracking data coming from Augmenta systems.
- Console, Mathias Markl. A console, which can be drawn to the screen.
- Drop, Ramin Soleymani. A processing library that lets you drag and drop objects such as files, images, bookmarks, or text into your processing sketch.
- Game Control Plus, Peter Lager. Use joysticks, gamepads, and other control devices in your sketch.
- GifAnimation, Patrick Meister (extrapixel). Play, import, and export gif animations. Render to GIF!.
- Grab, Taka Iwai, Roy Tatum. An easy-to-use PDF/jpeg exporter.
- IgnoCodeLib, Paul Hertz. Hierarchical display list for graphics and text with export to Adobe Illustrator 7.0 file format.
- ImageLoader, Mathias Markl. Simple to use API to load images from either Instagram, Flickr, Google, Giphy, Tumblr or your file system.
- Image Sequence Player, George Profenza. A simple library for loading, playing back and displaying image sequences.
- MQTT, Joel Gaehwiler. MQTT library for Processing based on the Eclipse Paho project.
- MindSet Processing, Jorge C. S. Cardoso. Allows you to use the NeuroSky Mindset brainwave sensing headset with Processing.
- MuKCast, Mathias Markl. A light-weight client/server library.
- NXTComm Processing, Jorge C. S. Cardoso. Allows you to control the Lego Mindstorms NXT robots.
- Novation Launch Controller client, Half Scheidl. Control your sketches using the knobs and pads from Novation® Launch Control®.
- OBJExport, Jesse Louis-Rosenberg. This is a simple library to export surfaces from processing as OBJ files. It is used the same way the PDF library is used.

- OOCSI for Processing, Mathias Funk. Processing client library for the OOCSI design middleware and Data Foundry access.
- SelectFile, Ostap Andrusiv. Android library which provides Dialogs for selectInput(), selectFolder(), and selectOutput() methods.
- Simple HTTP Server, Ramin Soleymani. A simple HTTP Server for Processing, that serves static files and simplifies HttpHandler.
- Simple Multi-Touch (SMT), Erik Paluka. Multi-touch prototyping and development made simple.
- Simple Receipt Printer, Gottfried Haider. Print on attached Thermal Receipt Printers.
- Simple Touch, Gottfried Haider. Touch events for Raspberry Pi and other Linux-based computers.
- Skweezee for Processing, Bert Vandenberghe @ eMedia Research Lab, KU Leuven. Create rich interactions based on physical squeezes.
- Tablet, Andres Colubri. A library for using pen tablets from Processing.
- Tramontana, Pierluigi Dalla Rosa. Prototyping and creative kit for multi-device interactions and experiences.
- VSync for Processing, Maximilian Ernestu. Will magically synchronize variables among Arduinos and your Processing sketch.
- Video Export, Abe Pazos. Simple video file exporter.
- Websockets, Lasse Steenbock Vestergaar. Creates websocket servers and clients, which makes it possible to communicate with the outside world including web sites.
- WootingKeyboard, Shinhoo Park @ KAIST Interactive Media Lab. This library can be used to analyze the raw data of the Wooting keyboard.
- proJMS, Hauke Altmann. Publish and consume text messages to create peer-to-peer communication between processing applications.

Animation

- Ani, Benedikt Gross. A lightweight library for creating animations and transitions.
- Green, Zacchary Dempsey-Plante. A gaming library that enables easy 2D game creation within Processing.
- Sprites, Peter Lager. Sprite control and animation for games and other graphic applications.
- Tilt for Processing, Nick Fox-Gieg. Reads Tilt Brush files in Processing.

Hardware

- Apple Light Sensor, Martin Raedlinger. Interface to the Light Sensor in MacBook Pro computers.
- Arduino (Firmata), David A. Mellis. Controls Arduino boards running the Firmata firmware.



- BlinkStick, Arvydas Juskevicius. Interface BlinkStick–smart USB RGB LED.
- DMX4Artists, Jayson Haebich. Easy-to-use library for controlling DMX lights with an FTDI USB controller.
- dmxP512, Henri DAVI. DMX output for enttec DMX USB PRO and lanbox LCE.
- Eye Tribe for Processing, Jorge C. S. Cardoso. A library to get eye gaze data from the Eye Tribe device.
- GaussSense SDK for Processing, GaussToys Inc.. Library to use the GaussSense in Processing.
- GazeTrack: Eye-tracking for Processing (Tobii EyeX and 4C), Augusto Esteves. Library that supports basic gaze tracking using various Tobii eye-trackers.
- HPGLGraphics, Ciaron Linstead. Write HPGL output from Processing sketches, suitable for sending to a pen-plotter.
- Hardware I/O, The Processing Foundation. Access peripherals on the Raspberry Pi and other Linux-based computers.
- Intel RealSense for Processing, Florian Bruggisser. Intel RealSense support for Processing.
- Ketai, Daniel Sauter. Android library for working with sensors, cameras, multi-touch, networking, Bluetooth, WiFi Direct, Near Field Communication, and SQLite.
- Leap Motion for Processing, Darius Morawiec. Library to use the Leap Motion in Processing.
- Myo for Processing, Darius Morawiec. Library to use the Myo in Processing.
- PS3Eye, Thomas Diewald. A PS3Eye library using libusb.
- PixelPusher, Jas Strong and Matt Stone. System for controlling an unlimited number of LEDs.
- Simple Phidgets, Shachar Geiger. A really simplified wrapper for using Phidgets in Processing.
- Skweezee for Processing, Bert Vandenberghe @ eMedia Research Lab, KU Leuven. Create rich interactions based on physical squeezes.
- Sudden Motion Sensor, Daniel Shiffman. Interface to the Apple Sudden Motion Sensor in MacBooks.
- Sweep for Processing, Florian Bruggisser. Scanse Sweep LIDAR API for Processing.

Sound

- Beads, Ollie Bown. A library for adding flexible real-time audio to Processing sketches.
- Cassette, Shlomi Ho. Implementation of Processing Sound APIs for Android.
- ComposingForEveryone, Guido Kramann. Gives support for applications in sound generation, simple web-cam-image processing, numerical simulation and–provided by examples–especially for algorithmic real-time composition of music.

- Loom, Cora Johnson-Roberson. Patterns that change over time, flexibly mapped to audiovisual output.
- The MidiBus, Severin Smith. Minimal MIDI library for Processing, no frills, no limitations.
- Minim, Damien Di Fede. An audio library that provides easy to use classes for playback, recording, analysis, and synthesis of sound.
- Pd4P3, Robert Esle. A real-time audio synthesis library of Pure Data's signal objects for Processing 3.
- ProcMod player, Arnaud Loonstra. An old-school MOD tracker player based on JavaMod!
- Sound, The Processing Foundation.Provides a simple way to work with audio.
- SuperCollider client for Processing, Daniel Jones. Framework to interface with the SuperCollider synthesis engine.
- tactu5, Alessandro Capozzo. Aids in the creation of algorithmic music in real time. It consists of a set of classes focused on defining musical elements, utility classes and an aggregator.
- ttslib, Nikolaus Gradwohl. Makes your sketches speak with the help of freetts.
- wellen, Dennis P Paul. Framework for exploring and teaching generative music making and algorithmic compositions.
- XYscope, Ted Davis. Library for Processing to render graphics on a vector display (oscilloscope, laser) by converting them to audio.

Video & Vision

- BlobDetection, Julien 'v3ga' Gachadoat. Computer vision library for finding blobs in an image.
- BoofCV for Processing, Peter Abele. Processing interface for BoofCV.
- GL Video, Gottfried Haider. Hardware-accelerated video on the Raspberry Pi & Linux.
- Image processing algorithms, Nick 'Milchreis' Müller. Implementations of basic image processing algorithms for processing.
- Kinect4WinSDK, Bryan Chung. A simple wrapper for the Microsoft Kinect for Windows SDK version 1.8.
- Kinect v2 for Processing, Thomas Sanchez Lengeling. Kinect v2 implementation using the Kinect Windows SDK.
- nyar4psg, R.Iizuka. NyARToolkit for proce55ing(NyAR4psg) is front-end of NyARToolkit for Java. This library can easily make the AR application.
- OpenCV for Processing, Greg Borenstein. Computer vision with OpenCV.
- Open Kinect for Processing, Daniel Shiffman. A Mac OS X Kinect implementation using open-source drivers (libfreenect).



- PS3Eye, Thomas Diewald. A PS3Eye library using libusb.
- Spout for Processing, Lynn Jarvis. For openGL texture sharing between Microsoft Windows applications using the Spout framework.
- tramontanaCV, Pierluigi Dalla Rosa. A toolkit for sensing people in spaces with phones.
- VLCJVideo, Caldas Lopes. VLCJ binding for Processing.
- Video, The Processing Foundation. GStreamer-based video library for Processing.

3D

- Camera 3D, Jim Schmitz. Alter P3D Rendering to produce Stereoscopic Animations, 360 Video and other 3D effects.
- Collada Loader for SketchUp and Blender, Markus Zimmermann. Importer for kmz and dae files created by 3D softwares SketchUp 8 or Blender 2.75a.
- extruder, Max Farrell. A 3D library to create extrusions.
- iGeo, Satoru Sugihara. 3D geometry library with packages of NURBS geometry, polygon mesh geometry, vector math, 3D display and navigation, 3D data file I/O and agent-based 3D geometry modeling.
- Lunar, Boy d'Hont. Parametric design library for the minimalist, inspired on existing node-based plug-ins for CAD software. Holds algorithms for easy generation and adaptation of polygon meshes, vectors, and list patterns.
- OCD: Obsessive Camera Direction, Kristian Damkjer. Allows intuitive control and creation of Processing viewport Cameras.
- Patchy, Jonathan Feinberg. Provides an easy-to-use bicubic patch for 3D Processing sketches.
- PeasyCam, Jonathan Feinberg. A mouse-driven camera-control library for 3D sketches.
- Picking, Nicolas Clavaud. Pick an object in a 3D scene.
- planetarium, Andres Colubri. This library provides a renderer to project 3D scenes on a full dome.
- QueasyCam, Josh Castle. A super-simple FPS camera for Processing.
- Shapes 3D, Peter Lager. 3D Shape creation and display made easy.

Math

- Combinatorics, Florian Jenett. Generate combinations, variations and permutations.
- grafica, Javier Gracia Carpio. Create simple and configurable 2D plots with Processing.
- gwoptics, Daniel Brown and Andreas Freise. Tools for drawing graphs in 2D and 3D.
- Jasmine, Peter Lager. A super-fast numerical expression and algorithm calculator.
- QScript, Peter Lager. Algorithm and expression evaluator.

Geometry

- Computational Geometry, Mark Collins & Toru Hasegawa. A simple, lightweight library for generating meshes such as isometric surfaces, boundary hulls, and skeletons.
- Dashed Lines, Jose Luis Garcia del Castillo. Draw shapes with dashed lines!
- Geomerative, Ricard Marxer. Extends 2D geometry operations to facilitate generative geometry. Includes a TrueType font interpreter.
- OCT, Thomas Wegener. A Processing library to create, modify, and display Octree structures.
- Squarify, Agathe Lenclen. A squarify treemap layout generator.
- handy, Jo Wood. Hand-drawn sketchy rendering in Processing.

GUI

- ControlP5, Andreas Schlegel. A GUI library to build custom user interfaces for desktop and Android mode.
- G4P, Peter Lager. Provides a set of 2D GUI controls and multiple window support.
- Guido, Florian Jenett. A simple cross-mode GUI library.
- Interfascia, Brendan Berg. Builds simple yet gorgeous user interfaces.
- meter, Bill (Papa) Kujaw. Display software, Arduino, or other sensor values in an analog meter.
- UiBooster, Nick 'Milchreis' Müller. Creates fast and easy GUI components for your sketch.

Language

- Eliza, Andres Colubri. The classic Eliza psychologist program.
- RiTa, Daniel C. Howe. A library for natural language and generative writing.

Typography

- Fontastic, Andreas Koller. A font file writer to create TTF and WOFF (Webfonts).

Compilation

- GenerativeDesign, Hartmut Bohnacker, Benedikt Gross. A collection of various functions belonging to the book Generative Design (English), Generative Gestaltung(-German) and Design Generatif (French).
- gicentreUtils, Jo Wood. Assists creation of data visualization sketches.
- ToxicLibs, Karsten Schmidt. An independent, open source library collection for computational design tasks.

Other

- Green, Zacchary Dempsey-Plante. A gaming library that enables easy 2D game creation within Processing.
- Path Finder, Peter Lager. Find paths through 2D/3D navigation graphs.
- Ptmx, Caldas Lopes. Add Tiled maps to your sketch.
- Steganos, Peter Lager. Steganography made simple.



## PRO 010 Processing People, 2001–2021.

This is a list of people who have joined the team to work on Processing or the Processing website. It doesn't include people who have created independent software libraries or who have moderated the Processing forums. We started this list in 2003 and have published it to the Processing website since. When people are no longer actively working on Processing, they are moved into the Alumni section.

### Project Leads

Ben Fry and Casey Reas started Processing in Spring 2001 and continue to obsessively work on it. In 2012, they started the Processing Foundation along with Dan Shiffman, who formally joined as a third project lead.

### Developers

- Andrés Colubri (Boston), OpenGL / Video
- Elie Zananiri (New York), Contributed Libraries / Tools
- Samuel Pottinger (San Francisco), Processing Core

### Libraries, Tools

The core Processing software is augmented by libraries and tools contributed through the community. These inventive extensions are a bright future for the project. We have a list of Contributed Libraries and Contributed Tools posted online. These contributions can't be underestimated. For example, see how Karsten Schmidt has transformed Processing through the toxiclibs library and how Damien Di Fede has extended the project into programming sound through the minim library.

### Website and Design

Design Systems International designed and built the current website and the new Processing family of logos through a year of dedicated volunteer work. A remarkable group of volunteers converted all of the content from the prior Processing website to the new formats. A hearty round of applause for:

- Tetsu Kondo
- Mark Webster
- Lionel Radisson
- Chris Coleman
- Justin Gitlin
- Seenahm Suriyasat
- Shobhit Sharma
- Karan Dudeja
- Mark Hancock

- Peter Jacobson
- Oğuzhan Göregen
- Bryan Ma
- Ashley James Brown

Processing People Alumni

- Jakub Valtar, Processing Core
- Scott Garner, Hello Processing Website
- Scott Murray, Website / Reference / UI
- Gottfried Haider, Processing for Pi
- Florian Jenett, Forum
- Jamie Kosoy, Website
- Manindra Moharana, PDE / Core
- James Grady, Visual Design
- Patrick Hebron, Video Library (Summer 2011)
- Peter Kalauskas, Library/Tool/Mode Install Utility (Summer, Fall 2011)
- Andreas Schlegel, Libraries (Winter 2008 - Summer 2011)
- Harshani Nawarathna, Processing Development Environment (Summer 2011)
- Cindy Chi, Reference Editing (Summer 2011)
- Jonathan Feinberg, Parsing & Android Hacking (Spring 2011)
- Chris Lonnen, Processing Development Environment (Summer 2011)
- Eric Jordan, Graphics Weapon (2007 - 2009)
- Tom Carden, Processing Hacks Director (Summer 2005 - Fall 2008)
- Lenny Burdette, Website renovation (Summer 2005 - Winter 2006)
- Simon Greenwold, Lighting and Camera (Winter 2005)
- Kevin Cannon, Website CSS (Fall 2004)
- Toxi, Graphics Gem (Summer 2003 - Summer 2004)
- Ariel Malka, Bagel Papa Poules (Summer 2003 - Winter 2004)
- Martin Gomez, Web engines (Spring 2003)
- Mikkel Crone Koser, Windows Platform Czar (Summer 2003 - Winter 2004)
- Koen Mostert, Windows Platform Czar (Summer 2003 - Winter 2004)
- Timothy Mohn, Platform Czar Mac (Winter 2003 - Winter 2004)
- Dan Mosedale, Preprocessor and Compiler (Spring, Summer 2003)
- Carlos Rocha, Sound and Graphics (Spring, Summer 2003)
- Jacob Schwartz, Windows Platform Czar (Winter 2003 - Winter 2004)
- Cem Uzunoglu, Web Scripts for Exhibition (Summer 2003)
- Dara Kilicoglu, Web Scripts for Exhibition (Summer 2003)



- Sami Arola, 3D Graphics Engine (Summer 2003)
- Marc Escobosa, Reference Engine (Spring 2003)
- Mathias Dahlström, Examples, Reference (Spring 2003)
- Dan Haskovec, Processing Environment (Summer 2003)

Alpha Reference Translators

- Widianto Nugroho, Indonesian
- Tetsu Kondo, Japanese
- William Ngan, Tori Tan, Mei Yu, Chinese Traditional and Simplified
- Art Center Nabi, Tae-Kyung Kim, Korean
- Julien Gachadoat, French
- Pedro Alpera, Spanish
- Alessandro Capozzo, Italian
- Burak Arikan, Turkish

We offer a special “Thank You!” to Sami Arola for writing the base of the original P3D and Simon Greenwold for incorporating camera and lights. Tom Carden contributed great energy by co-creating Processing Hacks and maintaining Processing Blogs. Andreas Schlegel did amazing work for over three years organizing the Contributed Libraries and building templates and documentation.

PRO 011

Processing and FLOSS, 2017,
Casey Reas.

Processing is *FLOSS* software. FLOSS is an acronym for *Free*, *Libre*, *Open-Source Software*. As defined by the Free Software Foundation, the term *Free Software* has four essential freedoms:

- to run the program
- to study and change the program in source code form
- to redistribute exact copies
- to distribute modified versions

The term *Open Source* originated after Free Software. It was defined to move the conversation toward one aspect of Free Software, the ability to read the source code. It's clear that the name Free Software leads to a partial understanding of what it means, that the software is likely to be "free" to use. This phrase Free Software is also confused with *Freeware*, another category of software where the software is free to use, but the source code can be closed. Many people in the Free Software community argue with people who promote Open-Source software about fundamental ideas and vice versa.

The acronym FLOSS is sometimes used to bridge the communities and their differing opinions. The word *libre* is added to *free* to make the goal of "free as in freedom," the ideals of liberty, more clear. Within the domains of computer science and systems administration, there are many influential FLOSS software projects. For instance, the Apache HTTP Server is used more than proprietary software in the same domain. In contrast, there's comparatively little FLOSS software within design and arts communities. These domains are dominated by proprietary software, specifically by Adobe Systems and Autodesk.

We believe visual artists and designers should be in control of their own software tools. It's not enough to rely on corporations to make what we need. The code and systems that we use shape our work. When companies release proprietary platforms and we use them, we don't have freedom to modify and expand them. As makers, we feel it's essential to have tools that we create ourselves, tools that are shaped by the needs and desires of the communities of creators that use them.

In addition to the creative limitations of proprietary software, there are also financial concerns. As I am writing this, a one-year license for the Maya animation software from Autodesk is $1,470 USD. A one-year license for the Illustrator drawing software from Adobe Systems is $240 USD, or $19.99 USD a month. In both cases, this is great for profits, but it's not good for access. This is beyond the reach of high school students and independent artists. It's also a large expenditure for educational institutions and students. In addition, the source code isn't available and the software can't be modified. There may be a place for proprietary, commercial software within the arts, but there's a need for alternatives to support wider and more diverse audiences.

All of the Processing Foundation software (Processing, p5.js, Processing.py, Processing for Android, Processing for ARM) are distributed without cost and the code is available for people to learn from and to modify. The Processing software is distributed under the GPL (GNU General Public License) and LGPL (GNU Lesser General Public License). Creating FLOSS software, however, is not without costs. We have learned, often the hard way, that "Free Software is expensive to make," but for people who want to use the software and who don't have the means to purchase or write their own software, ours is entirely free and open for use.

It's accurate to say that Processing wouldn't have launched without Free Software and other software under compatible licenses. Now, as when it was first released, Processing combines existing FLOSS software modules into its source code. These other projects include jEdit, ECJ, JNA, launch4j, quaqua, ORO, and Jikes. FLOSS software is an ecology where thousands of projects, large and small, can be combined in different ways to create new software. This creates complex contingencies. If one of these parts stops being maintained or has an error, the other software within the ecology are affected. There are negative impacts to building code within a FLOSS ecology, as well as the positives. As with everything, there are highs and lows to creating and working with Free Software. We've benefited tremendously from the work that others put into the software we utilize as core components of Processing and we hope that others have benefitted from our unique code. We believe in the fundamental freedoms of Free Software and that is our path.

PRO 012 Processing.py, 2010,
Jonathan Feinberg et al.

Python Mode for Processing was chiefly developed by Jonathan Feinberg, with contributions from James Gilles and Ben Alkov. The Python Mode examples, reference, and tutorials were ported and/or created by James Gilles, Allison Parrish, and Miles Peyton. Casey Reas, Ben Fry, Daniel Shiffman, and Golan Levin provided guidance and encouragement.

Support for the development of Processing.py came from many sources. Jonathan Feinberg implemented Processing.py independently from July 2010 through April 2014; since then, Google has kindly supported his efforts. In summer 2014, work on the Reference, Examples, and Tutorials was funded in part by the Integrative Design, Arts, and Technology (IDeATe) initiative at Carnegie Mellon University, and by a grant from the National Endowment for the Arts managed by the Frank-Ratchye STUDIO for Creative Inquiry at CMU. The Processing Foundation and Fathom have also provided critical logistical support for this work.

Much of the work in achieving compatibility with Processing 3.x was done by Luca Damasco (Google Summer of Code student), under the supervision of Golan Levin, with additional support from the Frank-Ratchye STUDIO for Creative Inquiry at Carnegie Mellon University.

Examples

```
Hue is the color reflected from or transmitted through an
object and is typically referred to as the name of the color
(red, blue, yellow, etc.) Move the cursor vertically over
each bar to alter its hue.
barWidth = 20
   def setup():
      size(32 * barWidth, 360)
      colorMode(HSB, height, height, height)
      WnoStroke()
      background(0)
   def draw():
      whichBar = mouseX / barWidth
      barX = whichBar * barWidth
      fill(mouseY, height, height)
      rect(barX, 0, barWidth, height)
      lastBar = whichBar
```



```
Continuous Lines. Click and drag the mouse to draw a line.
   def setup():
      size(640, 360)
      background(102)
   def draw():
      stroke(255)
      if mousePressed == True:
         line(mouseX, mouseY, pmouseX, pmouseY)
Conditionals are like questions. They allow a program to
decide to take one action if the answer to a question is true
or to do another action if the answer to the question is false.
   size(640, 360)
   background(0)
   for i in range(10, width, 10):
      # If 'i' divides by 20 with no remainder draw the first
      line
      # else draw the second line
   if i % 20 == 0:
      stroke(255)
      line(i, 80, i, height / 2)
   else:
      stroke(153)
      line(i, 20, i, 180)
```

PRO 013 Processing for Pi Technical Notes, 2017, Gottfried Haider.

The port of Processing to the ARM architecture has been developed by Gottfried Haider since 2015. It has been supported by a Google Summer of Code fellowship in 2015 and a Processing Foundation fellowship in 2017, both times mentored by Ben Fry. The website was developed by Maks Surguy, who also contributed the tutorials during his Google Summer of Code fellowship in 2018. Both greatly improved the project's accessibility.

The (original) Raspberry Pi contained an ARMv6 CPU, and 256 or 512 MB RAM. The Raspberry Pi 2 contains a quad-core ARMv7 CPU, and 1 GB of RAM. The Raspberry Pi 3 and 3+ contain a quad-core ARMv8 (64-bit) CPU, which can also be operated in an ARMv7 compatible mode. It contains the same 1 GB of RAM. The Raspberry Pi Zero and Raspberry Pi Zero W feature the same ARMv6 CPU as the original Raspberry Pi, and 512 MB RAM. They all contain the same Broadcom VideoCore IV graphics processor.

All models primarily run a modified version of the Debian Linux distribution named Raspbian that was made to run on the ARMv6 CPU (and higher).

On the Pi 2, 3, and 3+ it is also possible to run other, unmodified Linux distributions, such as Debian or Fedora, since those settled on the ARMv7 architecture as their "baseline" for modern ARM support. However on those distributions you might not have the necessary kernel modules and graphics library to make full use of the Pi's peripherals. This page thus specifically talks about running Processing on Raspbian.

"Legacy" Graphics

The Pi's graphics core exposes OpenGL ES 2.0, which is supported by Processing P2D and P3D renderer, thanks to specific enablement in the underlying library, JOGL. The graphics driver is built around a closed-source driver (found in /opt/vc), which limits our ability to troubleshoot bugs for the moment.

Due to a limitation of this driver, P3D is currently limited to using two lights. Certain sketches might run out of video memory and throw an exception mentioning GL_OUT_OF_MEMORY. You might be able to work around this by changing the memory split—the amount of memory allocated for the GPU from all system memory. To do so, open the Raspberry Pi Configuration (under Menu, Preferences), navigate to the Performance tab, change the amount of "GPU memory" and then restart your Pi.

Experimental Graphics

Alternatively, Raspbian also includes a free and open source Mesa driver, named vc4, which can be enabled by running sudo raspi-config in a terminal, and selecting either GL (Full KMS) or GL (Fake KMS) under Advanced Options and GL Driver. (The current default is Legacy, which is described in the section above.)



This driver might run notably faster than the legacy graphics, supports up to four lights, and does not show some glitches that plague the other driver.

As of now the experimental driver does not yet support hardware-enabled video decoding, the camera module, as well as some screens that can be attached to the DSI interface, which should be forthcoming in future versions of Raspbian.

GPIO

Keep in mind that the Pi uses 3.3V levels, rather than the 5V of the Arduino Uno. The pins are said not to be 5V tolerant, so make sure to keep your voltages to 3.3V.

Each pin is rated up to 16 mA per pin, with 50 mA total, across all pins. (The Arduino UNO is 20 mA @ 5V per pin.) Make sure not to draw more current.

The Hardware I/O library's GPIO class uses GPIO numbers with its methods. Those are not the same as the physical pin numbers of the pin header. (see pinout)

I2C

The Pi has one (publicly exposed) hardware I2C interface. To use it in Processing (with the I2C class in processing.io), open the Raspberry Pi Configuration (under Menu, Preferences), navigate to the Interfaces tab, enable I2C and then restart your Pi.

After restarting, I2C.list() should return one interface: e.g. i2c-1 on the Pi 2. The interface is located on pins 3 (SDA) and 5 (SCL) on the Pi's header. (see pinout) Ground is conveniently located right next to it, on pin 6. Use it together with the 3.3V supply on pin 1, since that is the level that the Pi expects.

SPI

The Pi has two hardware SPI interfaces, but which share all but the SS (Slave Select) pins. To use it in Processing (with the SPI class in processing.io), open the Raspberry Pi Configuration (under Menu, Preferences), navigate to the Interfaces tab, enable SPI and then restart your Pi.

After restarting, SPI.list() should return two interfaces spidev0.0 and spidev0.1.

The interfaces' pins are located on the Pi's header on pins 19 (MOSI), 21 (MISO), 23 (SCLK), 24 (SS, aka CE0) and 26 (SS, aka CE1). (see pinout) When using spidev-0.0, pin 24 (CE0) is being pulled low during a transaction, while pin26 (CE1) remains unchanged. When using spidev0.1, pin 26 (CE1) is being pulled low, while pin24 (CE0) remains unchanged. (This is to be able to address two devices on the same data & clock lines.)

LEDs

The Pi has two on-board LEDs, led0 and led1, which can be controlled through the LED class in Processing.

Since the regular user (named pi) is by default not permitted to write to the LED device, you must enable this once by running

```
sudo sed -i 's|exit 0|chmod -R a+rw /sys/class/leds/*\nexit
0|' /etc/rc.local
```

After a restart, the devices should be read- and writable by any user. (This can be confirmed by running ls -l /sys/class/leds/led0/brightness. The resulting line should start with -rw-rw-rw-.)

On the Pi, led0 is the green (I/O activity) light, while led1 is the red (power) light. They only can be turned on and off, so brightness() values besides 0.0 and 1.0 have no effect.

Serial

The Pi has one exposed serial port, on pins 8 (TXD) and 10 (RXD). (see pinout) Like all other pins, these operate on 3.3V TTL levels, instead of the RS-232 voltage levels normally expected from a computer's "serial port."

To enable the serial port device to be used with Processing, start the text-based Raspberry Pi Configuration tool by executing the following command in a terminal:

```
sudo raspi-config
```

With the arrow-keys and Enter, navigate to Interfacing Options, Serial. In the dialog that appears, answer No to the question whether or not to use the port for a login shell. Answer Yes to the question whether the serial port hardware should be enabled. Reboot the Raspberry Pi for the changes to have effect.

The serial port will be available to Processing's Serial library under the name /dev/serial0. (This will be an alias to /dev/ttyS0 on models that have Bluetooth functionality, and an alias to /dev/ttyAMA0 on models that lack Bluetooth.)

Video library

Use the new GL Video library to make use of the Raspberry Pi's accelerated video decoding hardware. (also available from the Contribution Manager)

Examples show the various ways the library can be used. Please file issues here.

*Video library: Capture*

If you're receiving the error IllegalArgumentException: No such Gstreamer factory: v4l2src with the (regular) Video library, try installing the necessary packages by executing sudo aptitude install gstreamer0.10-plugins-good in a terminal.

Alternatively, the GL Video library also contains some (very limited) functionality for using capture hardware. See this example for details.

If you want to use the Raspberry Pi camera with the GL Video library, add the following line to your /etc/modules file and reboot:

bcm2835_v4l2

(Note this is a lowercase L not a number one.) After the reboot your camera should show up as /dev/video0.



Touchscreen

The simpletouch library makes it possible to use any multi-touch-enabled display or trackpad with Processing, as long as the device is supported by the Linux kernel. This library is available through the Contribution Manager under the name "Simple Touch."

This works well with the official Raspberry Pi display, and allows for tracking of up to 10 fingers.

Two example sketches the library comes with explain how to use it. Please file issues here.

Libraries

Most libraries from the Contribution Manager work just fine without any change necessary to run on ARM. Exceptions to this are libraries that come with parts written in "native code," which is platform- and architecture-dependent, and hence needs updating. As a general rule of thumb: If you find (sub-) directories for different platforms inside the library's library directory, then this is likely the case.

If you come across a library that's not working, or if you need help compiling a library for ARM, please open an issue.

*Library: OpenCV*

ARM devices are supported by Greg Borenstein's OpenCV library starting with version 0.5.4 (available in the Contribution Manager).

*Library: OpenKinect*

A test version of Open Kinect for Processing, with support for armv6hf, can be found here. (PR #1 #2) On most ARM devices this will only work (if at all) with the Kinect 1, because of the high demand on USB throughput of the Kinect 2. Don't forget to place the file 51-kinect.rules in /etc/udev/rules.d for Processing to be able to access the Kinect's camera.

*Library: PureData*

The puredatap5 library allows you to write sketches in Processing that control and interact with musical patches prepared in Pure Data. See the accompanying HelloPd example for how it works. This library requires PortAudio to be installed, which seems to be the case for current releases of the Raspbian distribution. This library is not yet available through the Contribution Manager, but support for ARM was merged into its main repository.

*Library: Processing Sound*

ARM devices are supported by Processing's Sound library starting with version 1.4.0 (available in the Contribution Manager).

*FX2D*

The experimental FX2D renderer is not supported on ARM, because Oracle dropped support for JavaFX on ARM devices with Java 8u33. We might want to try using OpenJFX project in the future, but as of now this is unsupported.



PRO 014 The New Processing for Android Notes, 2017,
Andrés Colubri.

Processing's relationship to mobile devices goes back to the beginning. Ben Fry had versions working on Compaq iPAQs back in the early 2000s. Francis Li developed the Mobile Processing version of Processing to run as J2ME applications dating to 2005. The processing.org subdomain mobile.processing.org launched 20 September 2005!

The Android version of Processing recently reached an important milestone: version 4 of the Android mode, supporting the most recent releases of the Android Operating System, and enabling the creation of live wallpapers, watch faces, and Virtual Reality apps. Together with an updated site and an upcoming book, we are very excited to see what the Processing community will create with the new Processing for Android. This milestone would have not been possible without the help and support from many dedicated people at the Processing Foundation, Google Creative Lab New York, and Google Summer of Code program – the latter of which, specifically the projects of Sara Di Bartolomeo and Rupak Das, I will describe in more detail in this post.

Processing for Android has been around for a while. After initial conversations with Andy Rubin, the creator of Android, back in February of 2009, Ben Fry got Processing code from Casey Reas to work on the G1 phone, the first commercially released device to use Android. Ben and Jonathan Feinberg then wrote the foundations of the Android mode, which have been in use ever since. At the time, I had the opportunity to make my first contribution to the core of the Processing project by writing the 3D engine in Android mode, which eventually became the basis for the OpenGL renderer in the desktop version of Processing as well.

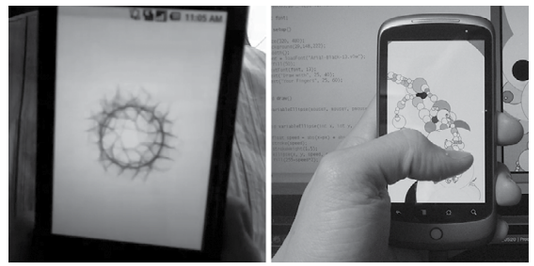


FIG 014–01 Left: Processing code running on the G1 phone in 2009. Right: Processing sketch running on the Nexus One in 2010.

However, the Android platform did not remain static. Since its first public release in 2009, it grew to become the most widely used mobile operating system worldwide, with many changes in its architecture and functionality along the way. With Processing for Android, we tried to follow up these changes to make sure that the Android mode is up-to-date and supports new devices and features. As I mentioned, the latest version of the mode lets you create VR and watch face apps, introduced in the past few years with Google VR and Wear.


FIG 014–02 Wrist watch with tree pattern filling the face.

It's worth noting how smartphones became the primary, if not only, computing device many people use on a regular basis. On one hand, it is natural to marvel at this feat of technology that put personal computers in the pockets of a significant percentage of the world's population. But on the other hand, we should ask ourselves how much control and ownership we have over the software that runs on these devices, and perhaps more importantly, on the data that they collect continuously about our interactions, habits, and whereabouts. A project like Processing can play an important role in answering these questions, by making the programming of mobile devices more accessible to their own users.

The open source aspect of Android has significantly helped us to develop Processing for Android, and to maintain it over the years. An important piece of this work is enabled by the Google Summer of Code (GSoC) program, which provides funding for college students from around the world to work on open source projects, paired with a mentor from the project. We were very lucky to have exceptionally talented students tackling challenges in Processing for Android, starting with Imil Ziyaztdinov and Mohammad Umair in the 2014 and 2015 editions of GSoC, respectively.

This year, we had two students working on Android-specific projects within Processing: Sara Di Bartolomeo and Rupak Das. What made their contributions very special is that they were key to this new release of the Android mode: Rupak worked on the underlying code inside the Android mode that does the heavy lifting by building the Processing sketches into Android apps, and updating the Android SDK. The transition from Ant to Gradle as the build system in the Android SDK posed a significant challenge. Since the Android mode originally used Ant, and Google removed Ant

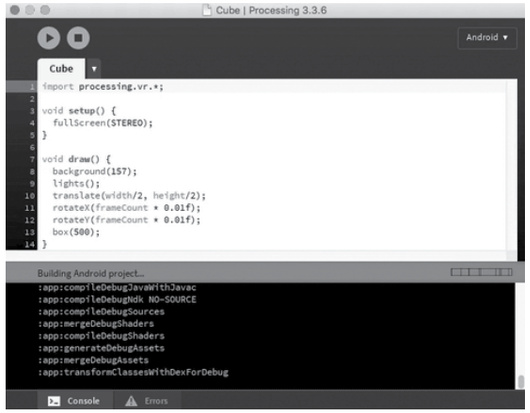



FIG 014–03 Processing editor with text "Building Android project..."



support from the SDK a few months ago, Processing for Android became incompatible with the latest versions of Android. Fortunately, Rupak was able to integrate Gradle tooling into the Android mode, which allowed us to move forward.

Sara, with mentorship from Gottfried Haider, complemented Rupak's work by creating a fully featured VR audio visualization that demonstrates what is possible with the new VR support in Processing for Android. Sara's VR Audioscape app combines FFT algorithms for real-time audio analysis with generative landscapes that represent the music track currently played on the device. All the source is available on GitHub for those interested in learning more about the techniques she applied in this project.

As Processing for Android marches into 2018 with exciting new possibilities, we invite you to explore creative and unexpected uses of our little pocket computers!

PRO 015

New Processing.org, 2021,
Design Systems International.

In time for the 20th anniversary of @ProcessingOrg, we are releasing a new visual identity for the Processing Foundation and its projects, and a new Processing website. Here's a quick look behind the scenes of our newest project.


FIG 015-01 Laptops with Processing logo stickers.


FIG 015-02 Tote bags with Processing graphics.

FIG 015-03 Buttons with Processing logos.

A new identity needs to support the existing community, while amplifying the message of accessibility. We speak to this in a language already set by Processing itself: a blend of art, play, and technology that's extremely friendly to new coders and familiar to longtime fans.

True to the spirit of the foundation, the visual identity is programmatic rather than static, and is centered around a program to derive logos from a set of simple rules: Draw three lines or curves in the intersections of an 8x8 grid.

The sea of potential logos turn into Rorschach inkblots eliciting everything from wiggly mascots to minimalist polygons. The logo system has the capability to produce endless icons across the foundation, be it logos for future projects, educational materials, or merchandise.

The logo family honors the history of the organization while bringing it up to date for its newest set of co-producers. With the grid as an invisible, unifying background, each project has its own recognizable identity without forgoing visual continuity.

In the Processing Foundation logo, one long squiggle connects two vertical lines in a purple hug. The logo puts across the spirit of collaboration and knowledge sharing across backgrounds, identities, and generations.



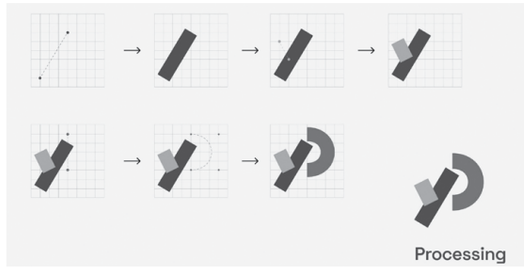



FIG 015-04 Vector diagram of Processing logo formation.

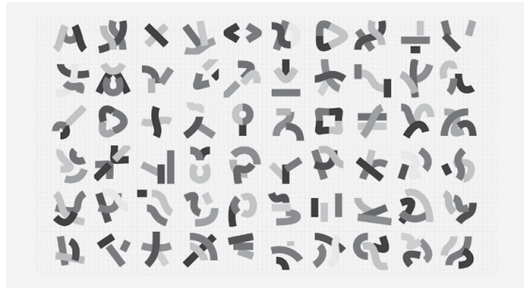


FIG 015-05 Grid of variety of Processing logo forms.

FIG 015-06 Two mugs with Processing graphic forms.

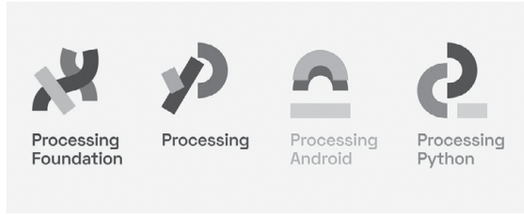



FIG 015-07 Processing Foundation, Processing, Processing Android, Processing Python logos with various curves.

The Processing logo balances legacy and a forward-thinking vision by reviving its former P logo with subtle tension and playfulness brought on by projecting it onto the grid.

The Processing Android logo uses the logo program to draw a visual connection with the classic Android character.

The Processing Python logo draws resemblance to a snake, loosely referencing the classic mirroring snakes in the Python logo.

Space Grotesk, an open-source font inspired by Space Mono, a typeface designed for writing code, is used throughout visual language. Space Grotesk was designed by the wonderful folks at @florian_karsten / http://instagram.com/floriankarsten

The Processing website is a crucial tool for community members as it hosts example projects, tutorials, teaching materials, and a reference library. We redesigned the website from the ground-up and did a complete overhaul of the content with the help of community volunteers.

Users are greeted with an interactive version of the logo program, projecting the same idea as that of Processing: technology is not merely passively received, but something everyone should be able to shape.

We unsheathed example sketches from behind their text descriptions and brought them to the Processing homepage. Visitors can click on each visual to

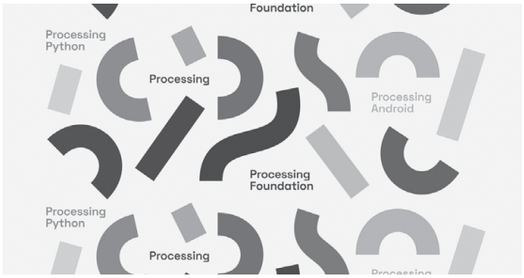



FIG 015-08 Decorative pattern with curves and logos.


FIG 015-09 Processing Foundation logo, three lines interweaving like a plant.


FIG 015-10 Processing logo, three curves forming a P.

view its interactive sketch and the code behind it. The result is image-forward, true to the software.

We improved both the accessibility and functionality of the reference pages by introducing a vertical layout with brief descriptions. A new sidebar with search makes it easy to navigate between pages and related examples demonstrate how to use the code in practice.

This project brings the Processing Foundation's ethos of accessibility to the forefront of its entire visual identity. We hope the new logo program, visual identity, and Processing website serve as a sincere thank you and a solid preparation for its next 20 years of growth.

We would like to thank @reas, @ben_fry, and the rest of @ProcessingOrg for trusting us with this project. We are always on the lookout for new collaborations, so get in touch if this project has piqued your interest!



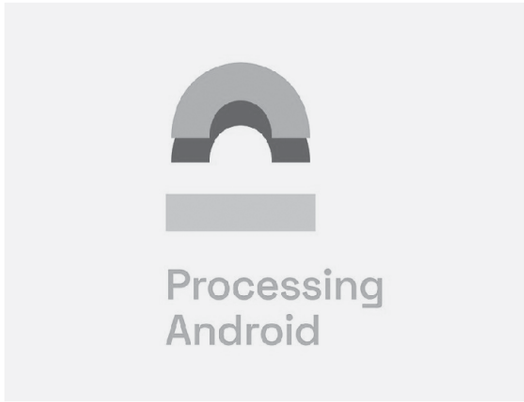



FIG 015-11 Processing Android logo, three curves forming A shape.


FIG 015-12 Procesing Python logo, three curves forming snake shape.


FIG 015-13 Series of PCD 2021 posters.

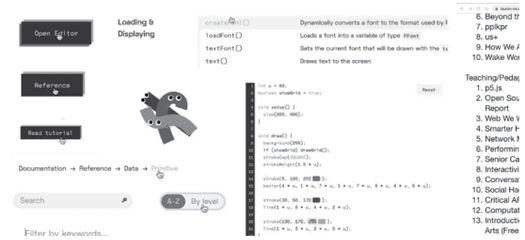



FIG 015-14 Screenshot of processing website with new logo.

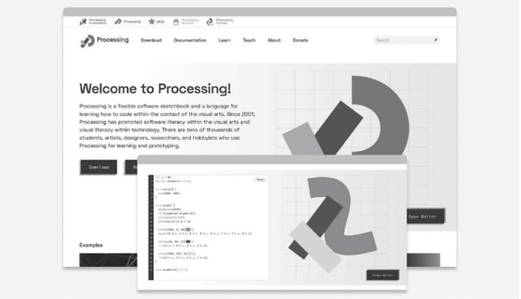



FIG 015-15 Processing website showing logo builder interface.

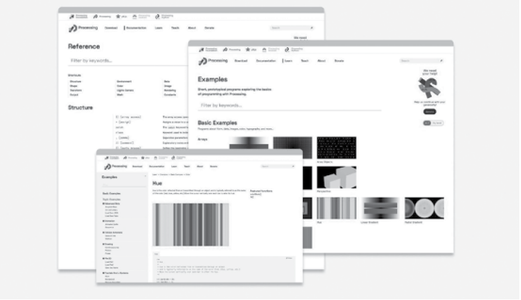



FIG 015-16 Processing examples pages.

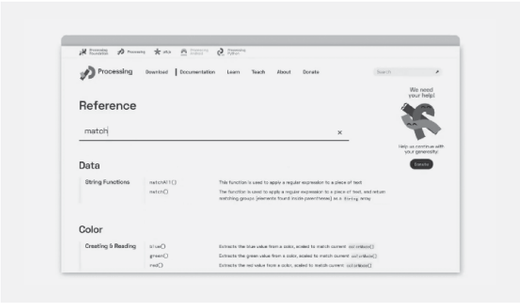



FIG 015-17 Processing reference page.


FIG 015-18 Processing application loading on laptop.

Initiated by Lauren Lee McCarthy in 2013 and developed by an international community of contributors, p5.js is a JavaScript library that aims to make creative expression and coding on the web accessible and inclusive. It uses the metaphor of “sketching with code”, drawing inspiration from its precursor, Processing. p5.js is free and open-source because we believe software, and the tools to learn it, should be accessible to everyone. p5.js is a community of, and in solidarity with, people from every gender identity and expression, sexual orientation, race, ethnicity, language, neuro-type, size, disability, class, religion, culture, subculture, political opinion, age, skill level, occupation, and background. We facilitate and foster access and empowerment. We are all learners.

P5* First p5.js Contributors Conference,
001 2015.

p5.js Contributors Conference 2015, May 25-31, 2015, Hosted by Frank-Ratchye STUDIO for Creative Inquiry, Carnegie Mellon University, Pittsburgh, PA.

FIG 001–01 A diverse group of participants smile and make the p5 sign with their hands.

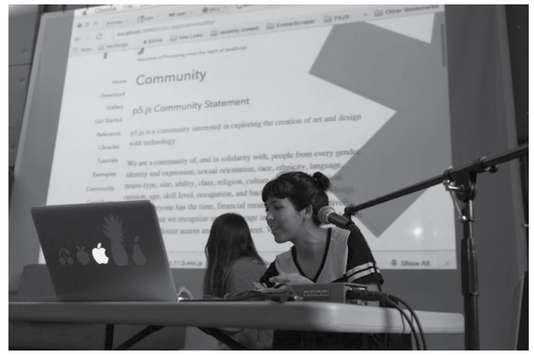



FIG 001–02 A person presenting the p5.js community statement from their laptop.

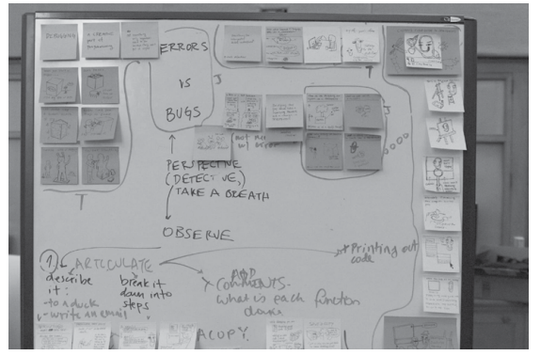



FIG 001–03 A whiteboard with different colored sticky and written notes about programming.

A group of approximately 30 participants spent a week at the Frank-Ratchye STUDIO for Creative Inquiry at Carnegie Mellon University in Pittsburgh, advancing the code, documentation, and community outreach tools of the p5.js programming environment. Participants came from as far away as Hong Kong, Seattle, Los Angeles, Boston, and New York. Most were working professionals in the fields of creative technology, interaction design, and new media arts, but the group also included a half dozen undergraduate and graduate students from Carnegie Mellon's School of Art.

Alongside technical development, one of the main focuses of this conference was outreach, community, and diversity. The conference began with a panel—Diversity: Seven Voices on Race, Gender, Ability & Class for FLOSS and the Internet (http://studioforcreativeinquiry.org/events/diversity-seven-voices-on-race-gender-ability-class-for-floss-and-the-internet). Organized by Johanna Hedva and Lauren Lee McCarthy, the panel took place Tuesday, 25 May 2015 in Kresge Auditorium at Carnegie Mellon University. Speakers included Maya Man, Casey Reas, Johanna Hedva, Stephanie Pi, Phoenix Perry, Taeyoon Choi, Sara Hendren, Epic Jefferson, and Chandler McWilliams.

FIG 001–04 Participants attentively smile and listen to a presentation.

FIG 001–05 Woman speaks at a podium in an auditorium while three participants sit on the stage and another three are Skyping in on the screen.

FIG 001–06 Five people in a circle with their laptops sharing their notes.

Participants included Jason Alderman, Sepand Ansari, Tega Brain, Emily Chen, Andres Colubri, Luca Damasco, Guy de Bree, Christine de Carteret, Xy Feng, Sarah Groff-Palermo, Chris Hallberg, Val Head, Johanna Hedva, Kate Hollenbach, Jennifer Jacobs, Epic Jefferson, Michelle Partogi, Sam Lavigne, Golan Levin, Cici Liu, Maya Man, Lauren Lee McCarthy, David Newbury, Paolo Pedercini, Luisa Pereira, Miles Peyton, Caroline Record, Berenger Recoules, Stephanie Pi, Jason Sigal, Kevin Siwoff, Charlotte Stiles.



P5* p5.js Community
002 Statement, 2015.

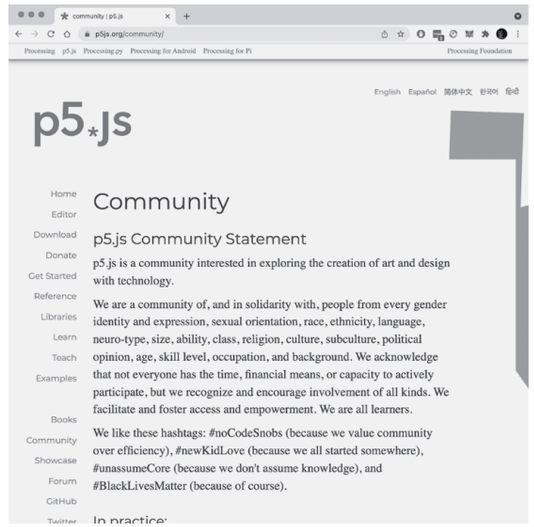



FIG 002–01 A screenshot of the p5.js Community Statement has black and pink text over a white background.

During the first p5.js Contributors Conference in 2015, we worked throughout the week to draft the p5.js Community Statement. The process incorporated ideas from all the individuals present, as well as other community members who participated online. The p5.js Community Statement has guided us through the years, shaping project decisions around design, code, documentation, language, teaching, organizing, and outreach.

p5.js is a community interested in exploring the creation of art and design with technology. We are a community of, and in solidarity with, people from every gender identity and expression, sexual orientation, race, ethnicity, language, neurotype, size, disability, class, religion, culture, subculture, political opinion, age, skill level, occupation, and background. We acknowledge that not everyone has the time, financial means, or capacity to actively participate, but we recognize and encourage involvement of all kinds. We facilitate and foster access and empowerment. We are all learners.

We like these hashtags: #noCodeSnobs (because we value community over efficiency), #newKidLove (because we all started somewhere), #unassumeCore (because we don't assume knowledge), and #BlackLivesMatter (because of course).

In practice

- We are not code snobs. We do not assume knowledge or imply there are things that somebody should know.
- We insist on actively engaging with requests for feedback regardless of their complexity.
- We welcome newcomers and prioritize the education of others. We strive to approach all tasks with the enthusiasm of a newcomer. Because we believe that newcomers are just as valuable in this effort as experts.
- We consistently make the effort to actively recognize and validate multiple types of contributions.
- We are always willing to offer help or guidance.

In times of conflict

- We listen.
- We clearly communicate while acknowledging others' feelings.
- We admit when we're wrong, apologize, and accept responsibility for our actions.
- We are continuously seeking to improve ourselves and our community.
- We keep our community respectful and open.
- We make everyone feel heard.
- We are mindful and kind in our interactions.

In the future

- The future is now.



## P5* 003 p5.js Friendly Error System, 2015–ongoing.

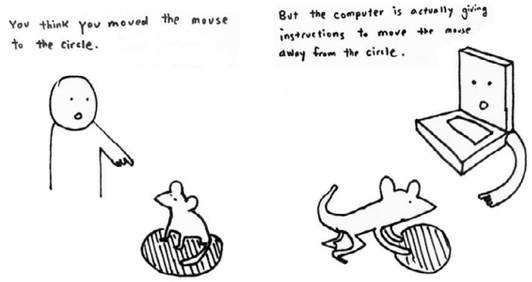



FIG 003-01 Image from the Field Guide to Debugging created by the Education Working Group, including Jason Alderman, Tega Brain, Taeyoon Choi and Luisa Pereira.

Hand-drawn comic showing a mouse moving a shaded circle.


FIG 003-02 Emily Chen presenting learnings from the PyLadies community at the p5.js Contributors Conference.

Woman sits with laptop in front of group and projection that reads "Don't assume we all should know X."

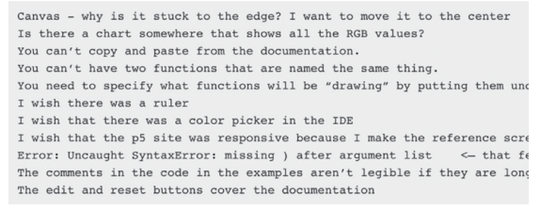



FIG 003-03 Jess Klein, beginning p5.js notes. Text notes identifying tricky parts of starting with p5.js.

The first version of the p5.js Friendly Error System (FES) was conceived and prototyped at the first p5.js Contributors Conference in 2015. The idea was to make the highly technical and often opaque error messages reported in the console more understandable to people using p5.js. The Friendly Error System would use simpler language, identify common errors, suggest changes to get the code working, and direct the user to documentation to learn more about the functions they're trying to use.

Jess Klein and Atul Varma expanded on this work in their fellowship project in 2016, using human-centered design principles to create more tools to help beginners get started with p5.js.

Alice M. Chung, mentored by Luisa Pereira, further developed the FES during Google Summer of Code in 2017. Alice gives an overview of the system below, excerpted from their project wrap-up essay (https://medium.com/processing-foundation/2017-marks-the-processing-foundations-sixth-year-participating-in-google-summer-of-code-d365f62fc463) published on Medium.

The voice of a debugger will have an impact on new developers' first impression and following coding experience. This is why designing a friendly debugger and error system is essential for projects with an outreach mission such as Processing and p5.js. Having a well-

designed debugger written in the right tone can lower the barrier for inexperienced users.

p5.js's new feature, the Friendly Error System (FES), was developed with this idea in mind. Starting from examining language-specific common error cases, we developed FES which creates console messages that can hint at the debugging solution rather than throw an error and exit. JavaScript often allows "silent failures," meaning the code will fail to execute the expected behavior without a warning (i.e., error messages) from the debugger. For example, JavaScript doesn't support type checking by default, making errors in parameter entry harder to detect for new JavaScript developers.

So far, FES is able to detect and print messages for two kinds of errors: (1) validateParameters() checks a function's input parameters to confirm it matches the inline documentation, and (2) friendlyFileLoadError() catches file loading errors. We have integrated these two kinds of error checking to a selected set of p5 functions, but developers can add them to more p5 functions, or their own libraries, by calling the above FES functions. FES provides a generalized error message generation system, so more error types can be implemented in the future.

To help beginners without requiring additional setup, FES is enabled by default in p5.js. In p5.min.js, it is completely disabled for efficiency. We understand that having an assistant system like FES can be counter-productive, and it is possible to easily disable FES by setting p5.disableFriendlyErrors = true.

Here are a few examples of the Friendly Error System in action, compiled by Alice:

File Load Errors

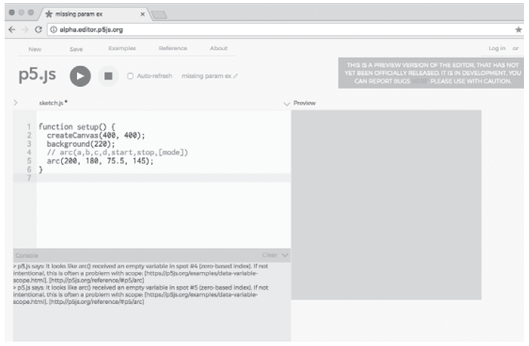



FIG 003-04 Screenshot of p5.js editor, friendly error message notifies the user the wrong number of arguments may have been entered.

Here is a simple scenario where I'm trying to stylize text using a font file, and the output fails to render the text in the desired style. In addition to JavaScript error messages, friendlyFileLoadError generates a message in the console:

> p5.js says: It looks like there was a problem loading your font. Try checking if the file path [assets/OpenSans-Regular.ttf] is correct, hosting the font online, or running a local server [https://github.com/processing/p5.js/wiki/Local-server].



The message first provides a short error case summary, and then displays the given path that seems to be incorrect and causing the problem. Lastly, it provides additional resources that may guide the users to a correct solution. This kind of file loading FES is currently implemented for the loadImage(), loadFont(), and loadTable() functions.

Missing Parameters

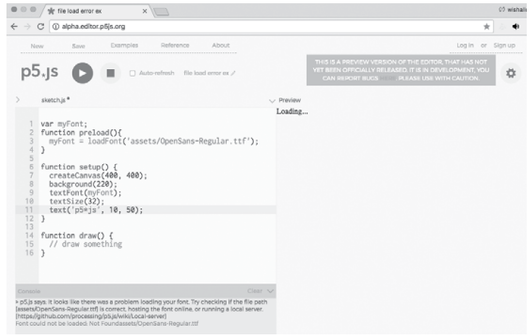



FIG 003-05 Screenshot of p5.js editor, friendly error message notifies the user there was an error loading font.

Here, I'm trying to draw an arc, but with a number of parameters less than the required. This is the silent error case that was discussed above, and thus no error from JavaScript. In the console, FES generates two messages, one for each missing parameters:

> p5.js says: It looks like arc() received an empty variable in spot 4. If not intentional, this is often a problem with scope: [https://p5js.org/examples/data-variable-scope.html]. [http://p5js.org/reference/#p5/arc]

> p5.js says: It looks like arc() received an empty variable in spot 5. If not intentional, this is often a problem with scope: [https://p5js.org/examples/data-variable-scope.html]. [http://p5js.org/reference/#p5/arc]

Similar to file loading error messages, the FES first provides a short error case summary with a location, and then provides additional resources that may be useful for debugging.

Wrong Parameter Type

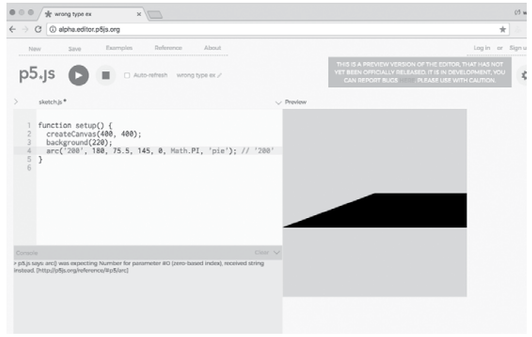



FIG 003-06 p5.js Editor friendly error flags arc has wrong parameter.

In a similar vein, FES also checks parameter types. In this scenario, I'm trying to draw an arc but gave the wrong parameter type String instead of the required parameter type Number. FES generates a message summarizing this with a link to the function's documentation:

> p5.js says: arc() was expecting Number for parameter 0, received string instead. [http://p5js.org/reference/#p5/arc]

Object Parameters and Multiple Parameter Formats (with an example for developers)

FES not only supports type checking for JavaScript's default types (Boolean, String, Number, Array, and undefined) but checks the object's name parameter.

Akshay Padte, mentored by Stalgia Grigg, added further improvements during Google Summer of Code in 2020. Excerpts from Akshay's project wrap-up essay (https://medium.com/processing-foundation/extending-the-p5-js-friendly-error-system-8af80dd2d314) published on Medium are included below.

The first improvement was to add a spellcheck kind of system to the FES. Beginners often need time to understand the various naming conventions commonly used in programming, such as camelCase for identifiers, CAPS for constants, etc. And so, capitalization and spelling mistakes are very common, such as writing create canvas() instead of createCanvas(), colour() instead of color(), etc. These kinds of mistakes are easier to resolve, as the browser would display an error pointing to the function call.

But if someone misspells a p5 entry point function (which has to be defined by the user), such as by defining preload() as preLoad()—which I learned is a very common mistake—p5 won't be able to detect it and the sketch would fail silently, and it may take a lot of time to debug this simple mistake.

Here are a few example messages from this feature:

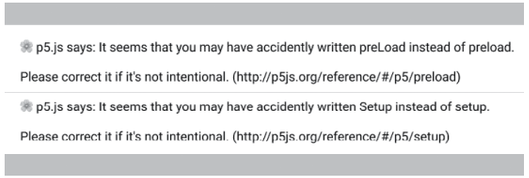



FIG 003-07 Examples of p5.js FES messages.

The second new feature was Global Error Catching. This meant analyzing the errors thrown by the browser and trying to match them up with helpful explanations to solve them.

To classify the errors, it was settled to use a regex match against a pre-built lookup table. The idea was based on the fact that web browsers use template error strings to generate error messages. This means that for a given browser, all error messages of a particular kind would have a consistent structure. We can have our own template strings with placeholders, which are then replaced with regex matching sequences (like ([a-zA-Z0-9_]+) ), and then the result is matched against the error message to detect what kind of error this is. The regex sequence also helps to extract relevant details. For example, take a look at this sketch:

```
function setup() {
   let a = 5;
}
```



```
function draw() {
   let b = a + 5;
}
```

It shows a very basic mistake with scope that a beginner can make. The error shown is:

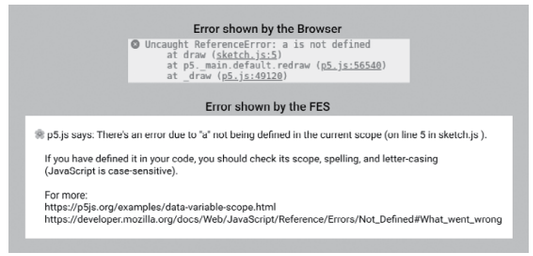



FIG 003-08 p5.js FES flags undefined variable.

While the browser error message aims at being concise, the FES message aims to explain the error as much as possible, and also provides links that have examples to fix this kind of error. This is more helpful to those who have just started learning to program and have not yet gotten used to deciphering error messages.

Another distinction to be made was between errors in user-space and errors that happened inside the library. These could be differentiated by seeing their stack trace. Moreover, it's possible to simplify the stack trace itself to only include user-defined functions.

Here's an example of an error that happens in library space:


FIG 003-09 p5.js FES filters out stack trace messages.

The FES filters out all the internal details from the stack trace, making it easier to understand.

Shantanu Kaushi, mentored by Thales Grilo and Luis Morales-Navarro, added new features during Google Summer of Code in 2021. Excerpts from Shantanu's project wrap-up report are included below.

FES's fesErrorMonitor has a list of errors that it detects. My work was to extend this list, enable fesErrorMonitor to detect more commonly seen errors, and show them in a simplified form.


FIG 003-10 A block of code that shows the error message: "p5.js says: You have used a p5.js reserved variable 'PI' make sure you change the variable name to something else," followed by a link to the relevant page in the reference library.

I added a new feature to FES to detect the redeclaration of p5.js's reserved variables and functions. If a user accidentally redefines a p5.js reserved constant/function, it can cause problems and create confusion. The new redeclaration detection feature tackles this problem.

P5* p5.js Accessibility Project,
004 2017–ongoing.

The p5.js accessibility is an ongoing project that began in 2017, with many people contributing their perspectives through testing, design, development, and documentation. Included below are excerpts from project write-ups from three key contributors to the project.

FIG 004–01 Image of workshop/focus group participants. Six workshop members sit around a table with laptops in front of them. They discuss and try a coding exercise.

p5 Accessibility, By Claire Kearney-Volpe, Originally posted on the Processing Foundation Medium 1 December 2017.

Over the last two years, a small community has been growing and working on the accessibility of p5.js. At its core, the p5 Accessibility project is a community-based attempt to make p5.js and its learning resources accessible to people that are blind or have low vision. The Processing Foundation has supported the work, which relies on a broad network of both sighted and blind experts and end-users.

The project started where I work at NYU's Ability Project, an interdisciplinary research and development space. In this studio, our faculty and student teams develop assistive and rehab technologies based on the premise that these technologies serve people best when they participate in its design. People from the community tell us their ideas for technological interventions, and we work with them to develop solutions. Although physically based in the Tandon School of Engineering, students from across NYU (engineering, occupational therapy, and design students) collaborate with end-users to develop assistive and rehab technologies.

One of these projects was a collaboration between Chancey Fleet and I, a blind technology expert and educator who works at a New York Public Library. Together, we worked to develop a diagramming tool that would allow her to plan a banquet, and save and share her plans digitally. The project ended up being a bit of a disaster.

Not only did we need to program a sonified, touch-screen, web application as coding novices, we also struggled to find cohesive, newbie-friendly documentation of accessible development practices. For me, there was also the challenge of learning how Chancey used her computer and cellphone. Many people who are blind, Chancey included, use screen readers with synthesized voices or braille displays that relay information about the content on their screens. Screen readers, it turns out, are not that easy to use (although worth a try).



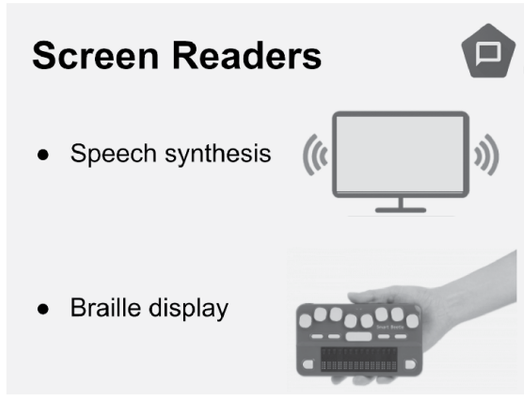



FIG 004–02 Diagram of Screen Readers. Diagram depicts representations of speech synthesis and braille outputs. Sound waves coming from a monitor indicate speech synthesis, and a small, red refreshable braille display is used as an example of screen reader outputting to braille.

In addition to being in unfamiliar territory, another factor impeded our progress. If the prototyping process was to be truly participatory, Chancey needed to brush up on JavaScript and HTML. We looked into the plethora of code-learning resources online and found that most were inaccessible to her screen reader. Even the "good" ones, the ones that teams of very smart people had worked hard to make accessible, remained unfriendly and cumbersome to use because although their lesson descriptions were accessible, their videos, edit fields, or screenshots of code snippets were not.

The lack of good documentation around accessible development practices and the lack of accessible code-learning resources slowed us down and frustrated us to the point that the project was stopped halfway. Although our banquet diagramming app was never fully realized, from this work emerged a sense of immediacy around accessible code learning.

Our solution was to combine the design challenges into one mega project. We approached the Processing Foundation, and they accepted our application to their 2016 fellowship program. The Processing Foundation's mission – to "promote software literacy" and "empower people of all interests and backgrounds to learn how to program and make creative work with code" – aligned nicely with the project goals.

We began the project with input from experts and focus groups that have facilitated a redesign of the p5.js learning resources and web-based code editor to ensure that they are accessible to people with low vision and blindness.

Expert Interviews:
We interviewed five professional HCI (Human Computer Interaction) researchers and developers with special interest in accessibility for people with visual impairments. Each of these stakeholders have visual impairments themselves, and offered great insights into the issues of this type of undertaking and examples of previous attempts.

Focus Groups and Code Learning Workshops:
We held five workshops in which people with visual impairments came and learned web development and also tested our p5.js web-editor prototypes as they have developed with feedback.

FIG 004–03 Student volunteer and participant during workshop.
Smiling student volunteer and participant sit at a desk with laptops in front of them.

FIG 004–04 Participants check on each others' work during coding exercises.
Claire stands behind three workshop participants sitting at a table with laptops in front of them.

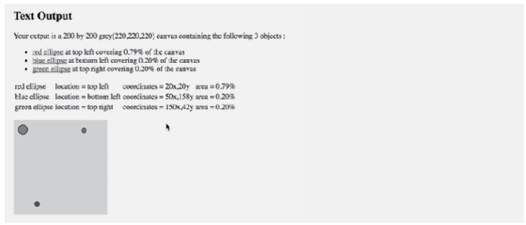



FIG 004–05 Visual example of text and table outputs of a simple sketch with animation.
Animated gif showing text and table outputs of a simple sketch with animation. Updating text and tabular data reads the position and size of ellipses in a canvas on the bottom left of the screen. A red ellipse is at the top left of the canvas and green and blue ellipses bounce vertically from top to bottom of the canvas.

Implementation:
Led by Mathura Govindarajan, Luis Morales-Navarro, and Cassie Tarakajian, we have been continually developing the accessibility of the p5 development environment (IDE) and its user interface elements/interactions across a variety of browsers, operating systems, and assistive technologies (AT). The outputs of p5 sketches have been a particular challenge to convey in an accessible format. As canvas elements are impenetrable by screen readers (the only HTML5 element that is), we worked on creating a "Shadow DOM" that dynamically interprets the visual and spatial outputs on screen. Our Shadow DOM outputs are hidden but accessible to screen readers.

The three outputs that we have developed are 1) accessible text (English language description), 2) tabular (in a table format), or 3) tonally based (panning and frequency used to convey position and speed) descriptions of the visual content on the canvas. In their current state, the accessible outputs handle fairly simple sketches and can be activated in p5's web-based IDE's Preferences menu.

We have also created a high contrast mode for the IDE and developed our own "Color Namer" which translates RGB values to color descriptions. With the assistance of Lauren McCarthy, we've worked on an audit and remediation of p5's resource website accessibility. This small but dedicated team is currently working on making an accessible widget




FIG 004–06 p5 accessibility-specific software work. Screenshot of high-contrast editor mode, screenshot of p5 website, and color namer logo.

for the editor, so that it can be embedded in online lessons and tested with high school students.

1 Brown, J. S., Collins, A., & Duguid, P. (1989). Situated cognition and the culture of learning. Educational researcher, 18(1), 32–42.
2 Bellotti, V., & Smith, I. (2000, August). Informing the design of an information management system with iterative fieldwork. In Proceedings of the 3rd conference on Designing interactive systems: Processes, practices, methods, and techniques (pp. 227–237). ACM.
3 Edwards, W. K., Bellotti, V., Dey, A. K., & Newman, M. W. (2003, April). The challenges of user-centered design and evaluation for infrastructure. InProceedings of the SIGCHI conference on human factors in computing systems (pp. 297–304). ACM.
4 Gerber, E. Conducting usability testing with computer users who are blind or visually impaired. (2002). CSUN Conference Proceedings, (Session# 189). Retrieved from http://www.csun.edu/~hfdss006/conf/2002/proceedings/189.htm.
5 Pea, R.D. Practices of distributed intelligence and designs for education. Gavriel Salomon. Distributed cognitions Psychological and educational considerations, Cambridge University Press, pp. 47–87, 1993.
6 Pullin, Graham. Design meets disability. MA: MIT Press, 2009.
7 Stefik, A., Hundhausen, C., Patterson, R. An empirical investigation into the design of auditory cues to enhance computer program comprehension. International Journal of Human-Computer Studies, 69, 12, 820–838, 2011, Elsevier.
8 Stefik, A., Siebert, S. An empirical investigation into programming language syntax. ACM Transactions on Computing Education (TOCE), 13, 4, 19, 2013, ACM.

Making p5.js Accessible, By Luis Morales-Navarro and Mathura Govindarajan. Originally posted on the Processing Foundation Medium 12 April 2018.
When we started working on the p5.js accessibility project the premise had been set: We were going to make the p5.js web editor accessible to people with visual impairments. Claire Kearney-Volpe's 2016 Processing Fellowship project began a conversation about the role the Processing Foundation should play in making its resources and learning tools accessible. What we mean by accessible is making the editor and p5.js resources usable with screen readers, and also making sure that the canvas is available in the form of text and sound. You can read more about how this project began here (https://medium.com/processing-foundation/p5-accessibility-115d84535fa8). Although we joined the project in late 2016, and are now working on it as a 2018 Fellowship project (with Claire as our mentor), we think of it as something that goes beyond Claire's and our fellowships, and ideally will be sustained as an integral part of the development of p5.js.

Together with Claire, 2017 PF Fellow Cassie Tarakajian (https://github.com/catarak), and in collaboration with community members, we have been working on the implementation of a high-contrast theme for the editor, text and table outputs for screen readers, and a sound output that visualizes movement. These developments

led to the creation of p5.accessibility.js (https://github.com/processing/p5.accessibility), a library that makes the p5.js canvas accessible to screen readers. As we make the library more robust and focus on creating accessible-for-all learning resources, it is a good time to reflect on what we have learned so far from the p5.accessibility.js project, what we plan to do in the future, and why making p5 accessible is important.

Why is it important to make p5 accessible?

When we talk about "coding for all" and "coding literacy," people who are visually impaired are most of the time left out [1]. Tools for programming have limited accessibility features and are generally directed toward computer science curricula. Accessible learning resources are also limited, with examples being displayed through screenshots or videos that are not screen-reader friendly. Similarly, in code editors most of the feedback is visual and cannot be understood through screen readers.

Because almost all of our communications are now mediated through computers, understanding computational media and computational thinking is essential, and the concept of coding literacy is more important and popular than ever before. Coding literacy is a gateway for thinking about the role code and computing play in our everyday lives and how code shapes the way we communicate [2]. Processing, the mother of p5, was created with the dual aims of demystifying programming and of approaching coding as a creative and exploratory process that should be accessible to all; p5.js continues these intentions.

Although this approach makes computational thinking accessible to more people, the visual nature of p5.js reduces its accessibility. Learners with visual impairments can express themselves visually and should be able to use p5 to engage with computational thinking. p5 makes text-based programming easier for beginners by allowing them to code in and for the browser; as Daniel Shiffman says, p5 "is great for actually doing stuff, but for learning, it's terrific" [3]. We think of p5 as a learning platform, which is why the development of p5.accessibility.js focuses on p5 for beginners.

Because of the lack of accessible learning resources, programming for learners who are visually impaired relies heavily on "self-taught" efforts [4], which forces them to seek support in online communities. Such communities, of the kind that Processing and p5 both nurture, are important because they provide support for learners in the form of online resources, forums, and tutorials, and challenge them to continue experimenting with their code [5]. As Andrea A. DiSessa writes in *Changing Minds: Computers, Learning, and Literacy*, "The more a community knows about something, the easier it is for each member to learn" [6]. Making our communities accessible can provide new learners who are visually impaired and "self-taught" learners with a vast network of resources, experiences, and access to the knowledge of others.



p5-accessibility.js

In August of last year we realized that making only the p5.js editor accessible wasn't enough, since everyone should have the ability to access p5.js sketches outside the editor. This became even more important when thinking about current learning resources available among the p5.js community. We attempted to adapt the interceptor created for the editor so that it could be used in the p5.js widget. Through this process we realized that the best approach was to create a library that could bring the accessibility features of the editor to any p5.js canvas.The accessibility library takes an "ability-based design approach" [7], which considers how systems can be designed to fit the abilities of whoever uses them, by providing different ways of exploring the canvas. It is not about providing a one-size-fits-all solution but about having an array of options that makes p5.js more usable for everyone, including but not limited to people with visual impairments and older users. Adding the library is easy and simple without requiring knowledge of object-oriented programming and allows users to choose which outputs they want to make available.

Challenges

Canvas

Trying to make the p5.js web editor and p5.js sketches accessible came with its own set of challenges, the main one being that the canvas is probably the single, most inaccessible element in HTML. While it's possible for a screen reader to recognize the element, it cannot identify all of the elements on it. The reason for this is simple: the canvas behaves just like a physical canvas, once you put paint on it, it covers up what's behind it. It's easy to see the color of each pixel but not understand how the pixels comprise shapes and elements. This poses a problem when you are trying to describe the elements on the canvas through code. To help solve this problem, Atul Varma introduced us to the concept of monkey patching (https://en.wikipedia.org/wiki/Monkey_patch). Monkey patching is essentially a way to piggyback existing code and add additional "features" to it. We added some code to the editor to make it work along with the library. When a p5 function is called, this code runs before the actual p5 function and makes a local copy of the parameters that the user passed into the function and the function itself. This, paired with the p5 function documentation, gave us a vast amount of information. For example, when a user types ellipse(50,50,20,20), the monkey-patch code will first look at this function and understand that the user is calling the function ellipse with the arguments 50,50,20,20. It will then look up this function in the documentation (https://github.com/processing/p5.js/blob/master/src/core/2d_primitives.js#L141) of p5.js to extrapolate that the user wants to draw an ellipse at coordinates(50,50) of height 20 and width 20. This information is stored locally and displayed

on the web editor in an accessible fashion (https://medium.com/processing-foundation/p5-accessibility-115d84535fa8#96ea), meaning that the visual canvas is broken down into text or sound and these descriptions can be made available to the user. This basic concept of monkey patching has slowly extended to include more details about the canvas such as: What color is an ellipse? Which corner of the canvas is it closer to? What percentage of the canvas does it occupy?

Once we had a substantial structure set up on the editor to be able to provide accessible outputs, the main issue was making sure that screen readers were compatible with the code editor. The most widely used screen readers today are VoiceOver (free with MacOS), JAWS (paid software for Windows), and NVDA (open-source software for Windows). While they all strive to be the best possible available software, due to the speed of software development cycles, they fail to keep up with the latest updates of web browsers. This means that we found ourselves in a place where we had to choose the "most compatible" code editor for all screen readers. After exploring the options with help from Cassie, we settled on using codeMirror (http://codemirror.net/). We still had the issue that certain features are not supported in some screen readers and browser combinations. (At the time, Edge had just been released, and they had openly announced that they did not support screen readers.) To avoid further confusion, we conducted a series of tests and settled on supporting the following browser + OS + screen reader combinations:

- Safari + VoiceOver in OSX
- NVDA + Firefox in Windows
- JAWS + Chrome in Windows

Colors

Out of all the issues we faced, the one that gave us the most laughs was colors. From the monkey-patch code, we'd get hex values or RGB values of colors and have to convert these values to names. While there are ample JavaScript libraries across the internet that do this, most of them include color names like "lazarin crimson," "bermuda gray," or "tickle-me pink." These names are entertaining, but not very helpful. A text output should not only be readable but also understandable [8]. We were looking for a library that gave us easy-to-understand color names – names like those of crayon colors we used growing up. We didn't find one so we decided to make it ourselves. We developed colornamer (https://www.npmjs.com/package/colornamer), a simple NPM module/JavaScript library that, when given a hex or RGB value, returns color names that are accessible to people who are blind or visually impaired.

Earlier this year, Claire and Scott Fitzgerald used the library in their programming classes with high school students who are blind at the Oysters and Pearls Tech



Camp, a learning center that integrates technology and science in schools that are inclusive to people who are blind. The results were promising. One student who is able to see colors at a short distance could not stop laughing when the color names the screen reader gave matched the colors he was able to see. For example, when combining red, green, and blue color values for the first time, he created and enjoyed the color "salmon pink."

What's next?

From its inception, the role of the community has been crucial in our decisions and in the design and refining of the tools. We continue to talk to community members, and blind and visually impaired experts who work with programming, to better understand the challenges we face and arrive at solutions together.

We're currently entering a phase of heavy testing to make sure that p5.accessibility.js is robust and fully integrated into the web editor. We are also working on making existing p5 learning resources accessible in order to make the Processing and p5 communities more inclusive. Our aim is to create a comprehensive collection of learning resources that users who have visual impairments can engage with.

We are grateful to assistive technology experts Chancey Fleet, Sina Bahram, and Josh Miele for their support, and to Cassie Tarakajian for the development of the p5.js web editor and working with us to make it more accessible. We are thankful to Atul Varma, Daniel Shiffman, Luisa Pereira-Hors, Scott Fitzgerald, and Johanna Hedva for their input. This project was born out of an initiative at the NYU Ability Project and is supported by a 2018 Processing Foundation Fellowship.

1 Richard E. Ladner and Maya Israel, "For all in Computer Science for All," Commun.ACM 59, no. 9 (Aug, 2016), 26–28. doi:10.1145/2971329. http://doi.acm.org/10.1145/2971329.
2 Annette Vee, Coding Literacy: How Computer Programming is Changing Writing (Cambridge, Massachusetts: The MIT Press, 2017).
3 Daniel Shiffman, Learning Processing: A Beginner's Guide to Programming Images, Animation, and Interaction, Second ed. (Amsterdam: Morgan Kaufmann, 2015).
4 Andreas M. Stefik, Christopher Hundhausen and Derrick Smith, "On the Design of an Educational Infrastructure for the Blind and Visually Impaired in Computer Science" (Dallas, TX, USA, ACM, 2011). doi:10.1145/1953163.1953323. http://doi.acm.org/10.1145/1953163.1953323.
5 Vee, Coding Literacy (2017).
6 Andrea A. DiSessa, Changing Minds: Computers, Learning, and Literacy (Cambridge, Mass.: MIT Press, 2000), 271.
7 Richard E. Ladner, "Design for User Empowerment," Interactions 22, no. 2 (Feb, 2015), 24–29. doi:10.1145/2723869. http://doi.acm.org/10.1145/2723869.
8 World Wide Web Consortium, "Web Content Accessibility Guidelines (WCAG) 2.0, Guideline 3.1" https://www.w3.org/TR/WCAG20/#understandable

## Accessibility Improvements for the p5.js Web Editor, By Cassie Tarakajian, Originally posted on the Processing Foundation Medium 12 June 2020.

The p5.js Web Editor (https://editor.p5js.org/) is a widely used open-source project with 200,000 active users a month. As the lead maintainer, I juggle many different tasks and priorities, and I organize my work by balancing maintenance tasks, such as responding to and organizing GitHub issues and projects, in which I focus on creating or improving one feature. I knew the web accessibility of the site needed some attention,

and I had wanted to make the space to improve it, but it felt overwhelming because I thought I didn't really know what I was doing. Fifteen percent of the world's population (https://qz.com/1407450/theres-already-a-blueprint-for-a-more-accessible-internet/) has a disability, such as low vision, blindness, hearing impairment, and deafness, yet web developers aren't typically trained to think about this population – I certainly was not. How could I learn to improve support for a huge number of people who use the web editor?

I'm inspired and humbled by the amazing work of folks in the p5.js community who have pushed the goal of accessibility forward. This includes Luis Morales-Navarro and Mathura Govindarajan, Claire Kearney-Volpe, and many others. They have patiently explained accessible design to me: how users with low vision and blindness should have an equally robust experience as sighted users, and how accessibility features should be on by default. I am also grateful for a grant from the Clinic for Open-Source Arts (COSA) (https://www.du.edu/ahss/opensourcearts/) at Denver University, which allowed me to focus specifically on accessibility.

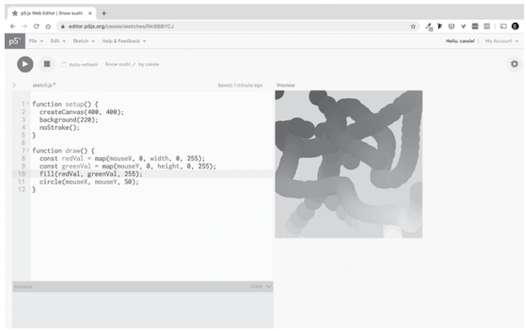


FIG 004-07 Screenshot of the previous light theme colors in the web editor.

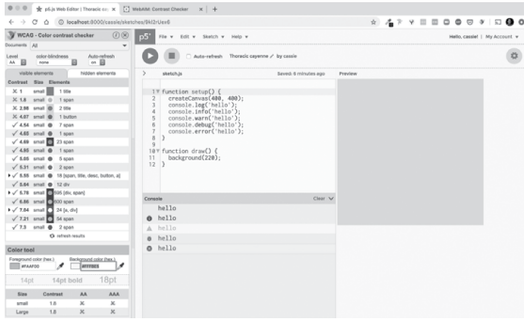


FIG 004-08 Using a color contrast checker to update low contrast colors in the web editor. Notice the updated higher contrast.
Contrast checker tool open in the browser.

The Web Content Accessibility Guidelines (WCAG) outline a thorough standard for web content accessibility, and they're fairly wordy (as standards should be!). Luckily, to make your website accessible, you don't need to read the whole thing, as there are many tools to help you align with these standards. There are specific criteria in three different levels (A, AA, and AAA) to meet folks with different needs, which cover different topics such as color contrast and semantic HTML.

I began with updating the editor. p5js.org website colors across the three different themes (light, dark, and high contrast). When teaching with the web editor, I had noticed that sometimes certain things were hard to see or read, especially on a projector. I used the web editor design system created by Jerel Johnson and a color contrast Chrome extension to painstakingly go through every element on the page. As a side effect, it helped me reduce the number of different colors used in each theme, clean up the styling code, and make some small improvements to the non-color interface design.

The pull request (https://github.com/processing/p5.js-web-editor/pull/1406) I made ended up fixing seven different open issues! It was very exciting. I even got to show off the crisp new colors in a livestream talk.

Next, I decided to tackle the icons, as there are a lot in the web editor: "play," "pause," "settings" cog, and so on. I knew that I needed to give them all labels that would be accessible to screen readers, but I didn't know how. I found and used "Contextually Marking Up Accessible Images and SVGs" (https://www.scottohara.me/blog/2019/05/22/contextual-images-svgs-and-a11y.html) by Scott O'Hara to guide me. I learned that sometimes icons convey information and need a label, and sometimes they are decorative and should be hidden from screen readers.

In order to add all of the necessary attributes to the SVG icons, I had to import the SVGs using a new library called SVGR, which imports all of the icons as React Components. As a side effect, it significantly reduced the number of network requests the web editor makes when loading, since the icons get bundled into the JavaScript.

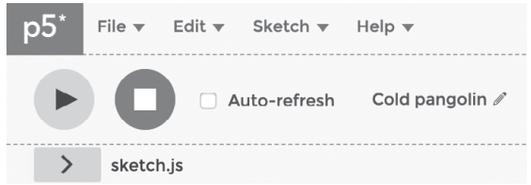



FIG 004-09 The arrow next to "File" is decorative, whereas the "play" icon conveys important information. p5.js editor toolbar.

Andrew Nicolaou (https://medium.com/processing-foundation/features-and-fixes-in-the-p5-js-editor-722e4b56495e), a Processing Foundation Fellow in 2017 and web editor contributor, had been working on using Storybook to build and document a component library to make contributing to the web editor interface easier. My changes to make the icons accessible caused merge conflicts with this branch, so I jumped in to fix these. In the process, I had the realization that by building a component library, accessibility features could be built into components so that you could use them without needing to fully understand them. This led me to create a Higher-Order Component for the SVG icons, based on what I learned from Scott O'Hara's post to account for decorative versus informational icons.

Lastly, I wanted to improve the semantic HTML (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/details), which adds meaning to your website content rather than change how it looks. Screen readers rely on semantic HTML to work properly, and by using the right HTML elements to match the meaning of your

```
function withLabel(SvgComponent) {
  const Icon = (props) => {
    const { 'aria-label': ariaLabel } = props;
    if (ariaLabel) {
      return (<SvgComponent
        {...props}
        aria-label={ariaLabel}
        role="img"
        focusable="false"
      />);
    }
    return (<SvgComponent
      {...props}
      aria-hidden
      focusable="false"
    />);
  };

  Icon.propTypes = {
    'aria-label': PropTypes.string
  };

  Icon.defaultProps = {
    'aria-label': null
  };

  return Icon;
}
```

FIG 004–10 Source code of React component with included accessibility features. Code in editor.

website content, you get many web accessibility features for free. For example, I knew that the web editor was missing <main> tags, which help screen readers skip navigation links at the top of the page and jump to the main content (which is easy to do visually). I added these to every page, and changed many s (which have no semantic meaning) to <section>s and <article>s. I even learned about HTML elements I had never used but would like to eventually, like <dialog> (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/dialog), which contains features that I'd engineered into the web editor from scratch.

There are still many accessibility improvements to be made to the web editor, but in only approximately 32 hours of work I made huge progress. The React Accessibility Guide (https://reactjs.org/docs/accessibility.html) is a great place to start if you're looking to make your website web accessible, even if you're not using React. Thanks to Chris Coleman and COSA for making this work possible!



## P5* 005 Diversity with Code + Art, 2017–ongoing.

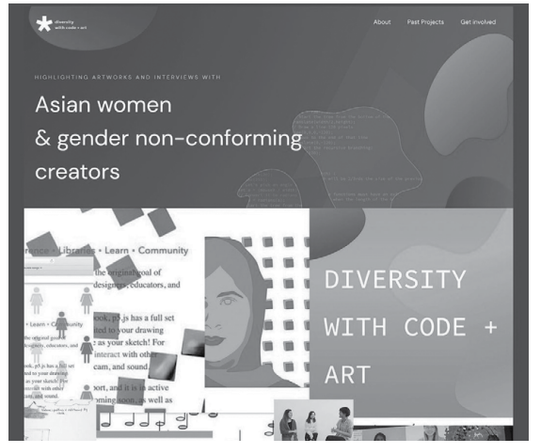



FIG 005–01 Asian women and gender non-conforming creators website with screenshots of p5.js sketches.

Diversity with Code + Art highlights diverse individuals amongst the art and code community, valuing representation and visibility. The project was created and led by Chelly Jin.

Series 1 (2017) highlighted Asian women and gender non-conforming artists, coders, and designers through a series of homepage artwork features and interviews. The series included Maya Man, Xin Xin, Wendy Tai, Qianqian Ye, Karen Peng, Yining Shi, Naomi Chan, Chelly Jin, and Haolin Fang.

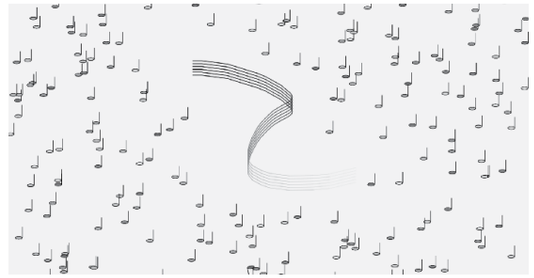


FIG 005–02 Maya Man. p5.js sketch with musical notes and stanza zigzagging.

FIG 005–04 Wendy Tai. p5.js sketch with fingers bending at different angles.

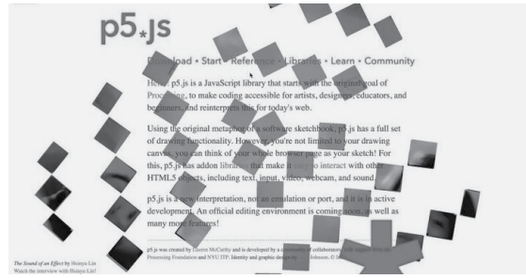



FIG 005–03 Xin Xin. p5.js sketch with grid of photos revealing webcam video underneath.

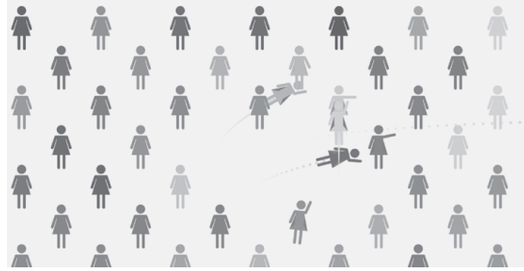


FIG 005–05 Qianqian Ye. p5.js sketch with grid of women flying like superheros.

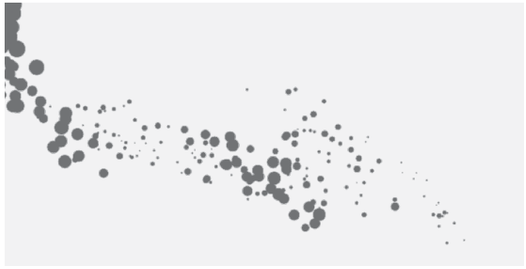


FIG 005-06 Karen Peng.
p5.js sketch with gray dots flowing from top left to bottom right.

FIG 005-08 Naomi Chan.
p5.js sketch with stroke lines creating triangular forms.

FIG 005-07 Yining Shi.
p5.js sketch with hexagons overlapping.


FIG 005-09 Chelly Jin and Haolin Fang.
p5.js sketch of Malala with mouths filling background.

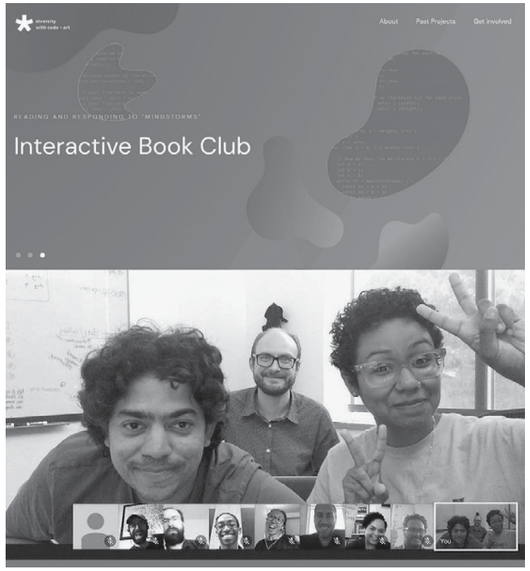



FIG 005-10 Interactive Book Club website with image from video chat with 12 people.

Series 2 (2017) was an interactive book club hosted by Sharon Lee De La Cruz, responding to *Mindstorms* by Seymour Papert.



P5* p5.js Internationalization, 2018.
006 The project of making p5.js more accessible to people who speak different languages began in 2015 when Maya Man created an alpha version of the p5js.org website with internationalization (i18n) features. The project was continued by Aarón Montoya-Moraga in summer 2016, who completed the i18n work of the website, making it possible to view the p5.js website and documentation in multiple languages. Since then, many people have contributed to the translation of p5.js materials, and have led teaching, outreach, and organizing efforts to bring p5.js to their local communities around the world.

At the Contributors Conference in 2019, an Internationalization Working Group developed the p5.js Global Contributors Toolkit, which included an overview of translation workflow, examples, and resources. It opened with this letter to global contributors:


FIG 006-01 Spanish translation of p5js.org. aarón montoya-moraga, Guillermo Montecinos, Website launched April 2018.
Two people stand in front of p5js.org website in Spanish.

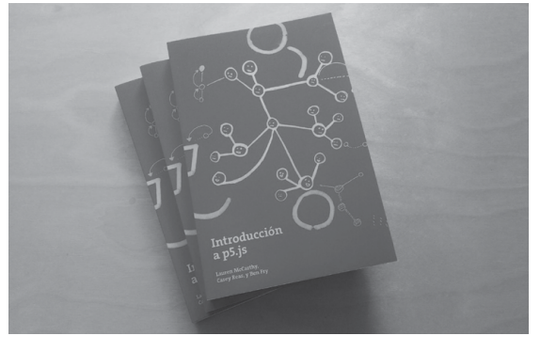



FIG 006-02 Introducción a p5.js. Translation of Getting Started with p5.js by aarón montoya-moraga, April 2018.
Stack of books with hand-drawn cover illustration by Taeyoon Choi.

First of all, we want to thank you for your interest in contributing to p5.js. We know how precious your time is and, with great appreciation, we welcome any kind of contribution that you could provide.

You don't have to be a software developer or a professional educator to contribute to p5.js. We celebrate contributions of any size and type from anyone who's excited about p5.js. We believe that contribution is about the process more than the product.

p5.js is not just a language, it's also a community. Since we're building a community, we encourage you to be mindful of your expectations and the responsibilities that come with them. We find that, often, taking small steps at the beginning would maximize your contribution in the long run.

You can help to build the p5.js global community in different ways:
• Participate in forum discussion.
• Create p5.js sticker in different languages.


FIG 006–03 Hindi translation of p5js.org. Sanjay Singh Rajpoot, Aditya Rana, Manaswini Das, Nancy Chauhan, and Shaharyar Shamsh.
p5js.org homepage with Hindi text.

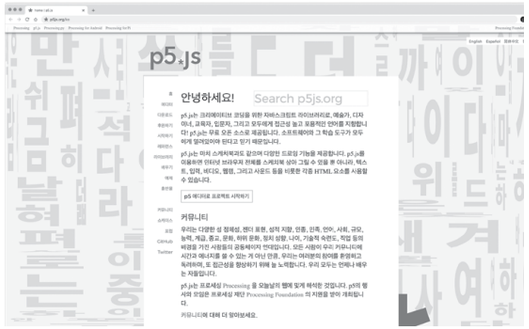



FIG 006–04 Korean translation of p5js.org. Inhwa Yeom, Seonghyeon Kim, Yeseul Song.
p5js.org homepage with Korean text, graphic characters in background.


FIG 006–05 Chinese translation of p5js.org. Kenneth Lim July 2018.
p5js.org homepage with Chinese text, mountain sketch in background.

- Create p5.js educational content in different languages.
- Contribute to the p5.js Github: report issue, pull/push request.
- Create zines about p5.js.
- Please add more.

It concluded with this note:
We want to acknowledge that JavaScript is an English-based programming language, upon which p5.js is developed. With that in mind, we understand the underlying colonialist/(neo)imperialist implications of making p5.js "globally available." We believe that English should not be the prerequisite to coding. Expressive coding is a form of creative expression regardless of one's background and language ability. Our translators have employed a variety of means to address this issue, such as rendering programming language into symbols and, in doing so, prioritize its function rather than its linguistic significance.
Of course, we are always finding ways to turn theoretical concerns into actionable steps.

Depicted in this section are some of the translation efforts thus far. We continue to work to make p5.js accessible in more languages.



P5* 007 Second p5.js Contributors Conference, 2019.

FIG 007–01 An overhead view of participants listening to a panel of people with an image of a 3D-rendered man on it.

FIG 007–02 A person sitting next to a lifesize teddy bear works on their laptop.

FIG 007–03 Group photo of participants smiling enthusiastically with their hands in the air.

An interdisciplinary group of 35 participants gathered at the Frank-Ratchye STUDIO for Creative Inquiry, advancing the code, documentation, and community outreach tools and exploring the current landscape of the p5.js programming environment. Comprising a diverse range of participants within the fields of creative technology, interaction design, and new media arts, the conference was aimed at fostering dialogue through a multidisciplinary lens. Working groups focused on several topic areas: Access; Music and Code in Performance; Landscape of Creative Tech; and Internationalization.

Organizers
Carlos Garcia, shawné michaelain holloway, Lauren Lee McCarthy, Luisa Pereira, Dorothy R. Santos, Tom Hughes.

Participants
American Artist, Omayeli Arenyeka, Sina Bahram, Aatish Bhatia, Natalie Braginsky, Jon Chambers, Luca Damasco, Aren Davey, Ted Davis, Carlos Garcia, Stalgia Grigg, Kate Hollenbach, shawné michaelain holloway, Claire Kearney-Volpe, Sona Lee, Kenneth Lim, Evelyn Masso, Lauren McCarthy, LaJuné McMillian, Allison Parrish, Luisa Pereira, Guillermo Montecinos, Aarón Montoya-Moraga, Luis Morales-Navarro, Shefali Nayak, Everest Pipkin, Olivia Ross, Dorothy R. Santos, Yasheng She, Jun Shern Chan, Cassie Tarakajian, Lauren Valley, Xin Xin, Alex Yixuan Xu, Qianqian Ye.

FIG 007–04 A group of people sits around an artificial campfire made from four LCD monitors.

FIG 007–05 A person with a microphone speaking to fellow participants.

FIG 007–06 Participants sit in a classroom facing the speakers listening intently.

FIG 007–07 p5.js Contributors Conference, Day 2, Computer Scientist and Scholar Sina Bahram presents his research on accessibility. The photograph is taken from an office area located above the conference meeting space showing participants seated around moveable tables, looking towards a large projection screen of the presenter's slides.

FIG 007–08 Participants conversing in a busy classroom.

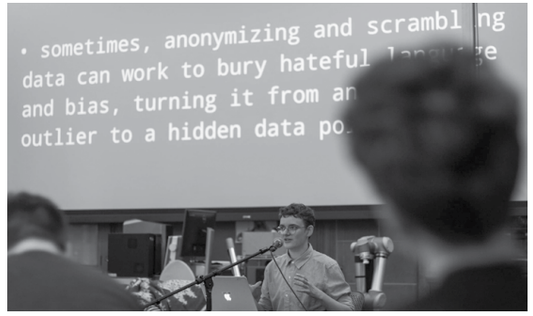



FIG 007–09 Participant speaks at a podium in front of projected text about the problem with anonymizing data.




FIG 007–10 Woman with microphone speaking to fellow participants with the text sacred boundaries in the projection behind her.

FIG 007–11 p5.js Contributors Conference, Day 1, Co-organizer shawné michaelain holloway speaking to participants. One of the co-organizers stands at a podium with a mic in her right hand while conference participants listen to her provide the schedule of events and activities for the p5.js Contributors Conference.

Outputs

- An introductory workshop on web accessibility led by Sina Bahram.
- A panel on Blackness and Gender in Virtual Space led by American Artist, with shawné michaelain holloway and LaJuné McMillian.
- New art installations by Stalgia Grigg, LaJuné McMillian, Aatish Bhatia, and Jon Chambers.
- A prototype of a notebook interface for p5.js. Created by Allison Parrish.
- A design of the p5.js library system for the p5 Editor. Created by Cassie Tarakajian and Luca Damasco.
- Prototypes connecting p5 to other libraries. Created by Alex Yixuan Xu and Lauren Valley.
- p5.js Global Contributors Toolkit. Created by Aarón Montoya-Moraga, Kenneth Lim, Guillermo Montecinos, Qianqian Ye, Dorothy R. Santos, and Yasheng She.
- How to write non-violent creative code. A zine led by Olivia Ross.
- An overhaul of the p5.js website for accessibility. Including updates for screen-reader accessibility, and improvements to the home, download, getting started, and reference pages. With contributions from Claire Kearney-Volpe, Sina Bahram, Kate Hollenbach, Olivia Ross, Luis Morales-Navarro, Lauren Lee McCarthy, and evelyn masso.
- p5grid. An implementation of highly flexible triangle, square, hexagon, and octagon girds for p5.js. Created by Aren Davey.
- p5.multiplayer. A set of template files for building a multi-device, multiplayer game where multiple clients can connect to a specified host page. Created by L05.
- Experiments using P5LIVE, testing early implementations of softCompile, OSC

FIG 007–12 Participants sitting at a long table having lunch and a discussion.

FIG 007–13 Participants in a meeting in a dark classroom.

interfacing and added connectivity with demo for MIDI setup. A p5.js collaborative live-coding vj environment! Created by Ted Davis.

- Collaborative performances by Luisa Pereira, Jun Shern Chan, Shefali Nayak, Sona Lee, Ted Davis, and Carlos Garcia.
- A performance by Natalie Braginsky.
- Workshops led by Everest Pipkin and Jon Chambers.
- A closing campfire circle led by Golan Levin.

Support

Our contributor conference took place at the Frank-Ratchye STUDIO for Creative Inquiry at Carnegie Mellon University, an academic laboratory for atypical, anti-disciplinary, and inter-institutional research at the intersections of arts, science, technology, and culture.

This event was made possible by a grant from the National Endowment for the Arts, and generous support from the Processing Foundation, Mozilla Foundation, Clinic for Open-Source Arts (COSA) at the University of Denver, NYU Tandon IDM, NYU ITP, FHNW Academy of Art and Design, DePaul University College of Computing and Digital Media, and Parsons School of Art, Media, and Technology at the New School.



## P5* 008 p5.js Access Statement, 2019.


FIG 008–01 Person with a microphone speaking to fellow participants in front of text that reads p5.js will not add any new features except those that increase access.

At the 2019 Contributors Conference in Pittsburgh, we asked how p5.js could further its commitment to access. Many conversations took place that week which were filled with care, wisdom, and hope. Six months later, Luis Morales-Navarro, Kenneth Lim, aarón montoya-moraga, Dorothy R. Santos, Johanna Hedva, Lauren Lee McCarthy, and I drafted this larger statement to record those ideas and subsequent agreements.

The primary goals were to provide clear criteria for what we would prioritize, while also inviting community members into a longer conversation about access in p5.js. We wanted to set a general direction and create space for more ideas. We thought it was important to offer specific examples of whose access we wanted to prioritize, and highlight the projects within the p5.js community that were already increasing access. As the statement says, this work was already being done, we just wanted more of it.

I felt intimidated by the prospect of writing this document. I hope sharing some of the challenges will help others feel less intimidated writing things like this in the future. Most of these are visible in the comment history of the GitHub pull request (https://github.com/processing/p5.js/pull/4676) which added the statement, but I'll summarize a couple here.

Some parts of creating this document were challenging. We were never satisfied with all of the language in the statement, especially when naming the people we were prioritizing access for. We used the term "marginalized people" in the opening paragraph. Nothing else we considered seemed significantly better, though we did choose "marginalized" over "underrepresented," "underserved," and "minority." Hopefully naming specific groups later in the statement helps clarify what we mean here.

For the list of groups that "Access here means making p5.js better for," we brainstormed ways to communicate something like "people from non-European/North American geographical locations." We considered terms like "non-Western" and "Global South," but they felt too reductive or wouldn't translate well into other languages. That list is not meant to be definitive (there are many groups that could be added), but we also left out some ideas for lack of sufficient terms.

This statement is the beginning of a conversation. Though it may be frozen on printed paper or a rendered PDF when you read this, we welcome your thoughts on how we can grow and shape access to p5.js. I'd love to see us revise this statement over time, or write something new to better capture our community's values and ways of working together.

evelyn masso.

## Our Focus on Access

At the 2019 Contributors Conference, we made a commitment to only add features to p5.js that increase access (meaning inclusion and/or accessibility). This means considering the vectors of diversity (e.g. gender, social, economic, race, ethnicity, language, disability, etc.) that can impact access/participation; and taking action to acknowledge, dismantle, and prevent barriers. We prioritize the needs of historically marginalized groups over the continued comfort of more privileged groups with p5.js.

We will not accept feature requests that don't support our effort to increase access. You'll see this criteria reflected in our issue and pull request templates.

This is part of an ongoing conversation about access and inclusion within p5.js. Our intention to hold these as core values from which p5.js is built is laid out in our Community Statement, which was written at the 2015 Contributors Conference.

**Please consider this a starting point.** We want to invite more conversations about what access means and how we can prioritize it.

### Kinds of access

Increasing access is not a focus on expanding the raw number of people in the p5.js community. It is a focus on making p5.js available to and approachable for people who are excluded from the p5.js community (intentionally or not) and from similar tools and communities.

Access here means making p5.js better for:

- People who speak languages other than English
- Black people, Indigenous peoples, and People of Color
- People who are lesbian, gay, bisexual, trans, or queer
- People with marginalized genders
- People with disabilities or illness
- People who lack opportunities and/or resources to engage with creative coding due to class or income
- People with little or no prior experience in open source and creative coding
- and other people who are systemically excluded and historically underrepresented

### Examples

These are examples of efforts we believe to increase access:

- Translating more documentation and other materials into more languages
- Improving our support for assistive technologies (such as screenreaders)
- Following Web Content Accessibility Guidelines in our tools and working towards making it easier for users to follow them in their projects
- Making p5.js error messages more helpful and supportive to people using the tool
- Mentoring and supporting learners of p5.js within communities that are historically excluded from and marginalized in creative coding and the digital arts

There are other things we haven't thought of yet and we're excited to figure out what they are together. If you have an idea, please share it as an issue.

FIG 008–02 Screenshot of "Our focus on access" document found in p5.js GitHub repository.



P5* 009 p5.js 1.0 Release, 2019.

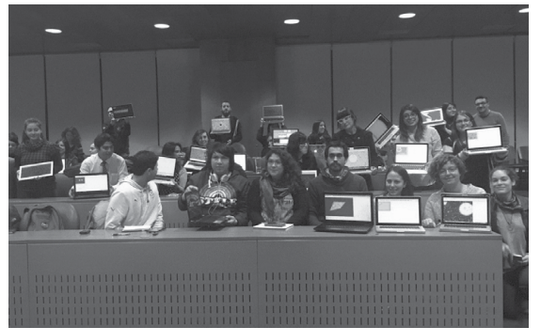


FIG 009–01 Artists in Santiago, Chile learn p5.js in workshop led by Aarón Montoya-Moraga Large group of smiling artists display their laptops with first colorful sketches made with p5.js.

Puedes leer la versión en español de este artículo aquí (https://medium.com/@ProcessingOrg/p5-js-1-0-est%C3%A1-aqu%C3%AD-42344aa2b4fd). Você pode ler a versão em português deste artigo aqui (https://medium.com/processing-foundation/chegou-p5-js-1-0-cb0647f632a5). 日本語版はこちら (https://medium.com/processing-foundation/%E3%81%8A%E3%81%BE%E3%81%9F%E3%81%9B-p5-js-1-0-%E5%85%AC%E9%96%8B-f8fb9bf1a734)です！

Today we are excited to announce the 1.0 Release of p5.js! p5.js is a JavaScript library which aims to make creative expression and coding on the web accessible and inclusive for artists, designers, educators, and beginners. While it's been nearly seven years since p5.js began, we intentionally embarked on the path to reach 1.0 a year ago when Kate Hollenbach worked on a first version of a roadmap for this. From there, the effort was led by Stalgia Grigg and evelyn masso, working with Lauren Lee McCarthy, Cassie Tarakajian, Kenneth Lim, and the thousands of contributors from around the world who joined in working on everything including code, documentation, teaching, outreach, writing, art making, and more. Reflecting the p5.js project values, 1.0 is not just a code-based milestone, but one that is grounded by significant work on the documentation and community.

Library Overhaul

In the past year working toward 1.0, we have put out five releases representing 1,488 commits (each commit can be thought of as a single round of changes in one or more files). You can download the new release at p5js.org. There were so many new things, we tried to capture key features and changes below. If you contributed something we missed here, please let us know at hello@p5js.org and we'll add it. :)

- Support for animated GIF drawing using the image() function!
- Addition of beginner-friendly drawing methods like circle() and square().
- Support for (and requirement of!) Alt-text within the image drawing methods. alt text, or alternative text, is the written text that appears in place of an image on a webpage if the image fails to load on a user's screen. This text helps screen-reading tools describe images to visually impaired readers.

- Updating all user-facing materials, contributor-facing materials, and the entire codebase and accompanying build processes to use ES6, led by Hirad Sab.
- Introducing new tooling into the build process to ensure maintainability of the code and prioritize accessibility, which includes things like linting and HTML validation to meet WCAG specifications.
- An improved p5.Serial library created by Shawn Van Every, Jen Kagan, and Tom Igoe that enables communication between p5.js sketch and Arduino (or other serial enabled devices).

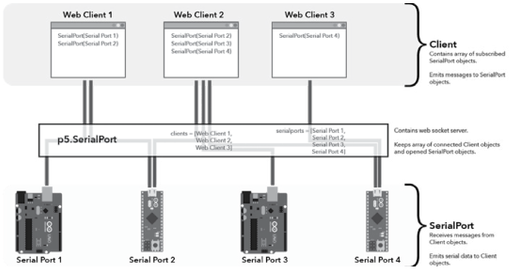



FIG 009-02 p5.SerialPort library overview. A diagram of Arduinos connected via p5.SerialPort library to web client windows.

- Updates to audio/video functionality and the p5.Sound.
- Library led by Jason Sigal to account for new browser requirements.
- Friendly Error System (FES) improvements. The FES is a system that beginners can turn on in the p5.js library which checks argument types, catches common errors, and provides more accessible explanations for how to fix the code in the console. We updated this to provide more helpful and intuitive friendly errors across the entire library.
- Addition of internationalization support for friendly error messages.
- WebGL drawing mode robustness. This included cleaning up text rendering, geometry drawing, and lighting; improving texture mapping capabilities; and simplifying and documenting the entire WebGL graphics pipeline.
- Merging our external DOM library into core to enable a range of functionality using HTML elements such as webcam, microphone input, video, audio, input elements, and file selection.
- Numerous bug fixes and documentation improvements in all areas.
- Review, simplification, and documentation of the build release process and tooling for future sustainability.
- Completion of comprehensive unit tests across the entire library to ensure the code keeps working with new changes.
- Implementation of GitHub bots, actions, and templates, including a friendly newcomer welcome bot and issue templates to aid new contributors.
- A new feature that enables contributors to initiate Twitter polls from GitHub issues to lower barriers to access for broader community participation and discussion.
- A redesigned set of contributor docs and a contributor docs website which document



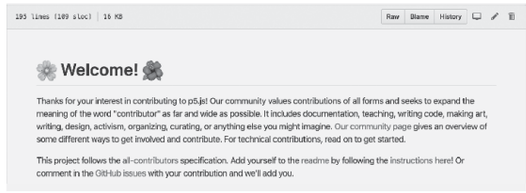



FIG 009-03 Screenshot of contributor docs welcome readme file. Screenshot of readme that says Welcome! with flowers around it, full text in readme.

how the project is organized and governed, and how to participate.

p5.js Editor

Through all this work, the p5.js Editor, led by Cassie Tarakajian, has been key in helping people of all ages and abilities to quickly get started creating, editing, and sharing p5.js sketches. The editor officially launched a little over a year ago and has continued to grow since then, recently passing over one million sketches created on the platform!

p5.js Contributors Conference

One of the key steps toward this release was a p5.js Contributors Conference held in August at the Frank-Ratchye STUDIO for Creative Inquiry at Carnegie Mellon University in Pittsburgh. We welcomed an extremely energetic, diverse, and generous group of people ranging from long-time contributors to those entirely new to the project. Working groups focused on several topic areas: Access; Music and Code in Performance; Landscape of Creative Tech; and Internationalization.

We spent significant time at the conference talking about the future of p5.js, especially sustainability and governance. Together, we came to the decision to explore a rotating model of leadership that would open the project to new perspectives and directions. It would also make the act of leading less burdensome, reducing barriers to entry. With this decision, it became clear that we would need to put significant effort into documentation and infrastructure to enable smooth transitions between leaders.

Documentation

Based on the conference and online discussions, we worked to document the project, its organizational and governance structures, and the various ways to contribute. Through these documents, we surface key ideas of structuring our project around community-building, diversity, and inclusion. The documentation can be found in a few different places:

- p5.js Website – The website structure and language were updated to be more intuitive and beginner-friendly. We also overhauled the entire site to be more accessible and WCAG compliant. This included adding tools to the website build process that will validate HTML pages and alert us to accessibility issues.
- p5.js Reference and Examples pages – Comprehensive and friendly documentation is a key aspect of p5.js. Reference entries and examples were added and updated to make functionality clearer and easier to learn.

- Contributor Docs – We worked on a folder of contributor documentation which guides people on things like getting started, navigating repository structure, adding documentation, creating libraries, understanding release processes, decision making, benchmarking, testing, and more.
- p5.js Global Contributor's Toolkit – A guide for international contributors and a reflection on what it means to contribute to p5.js. This addressed both the opportunities and problems, including the underlying colonialist/(neo)imperialist implications of making p5.js "globally available."
- How To Write Non-Violent Creative Code – A zine reflecting on the larger landscape of creative code, and how to embrace radical inclusion, decolonization, and a decentering of dominant communities within these projects and communities.


FIG 009-04 Screenshot from Calligraphy Brush Coding Challenge.
A screenshot that shows Qianqian at right with an insert of her screen, showing code in p5.js that produces Calligraphy Brush.

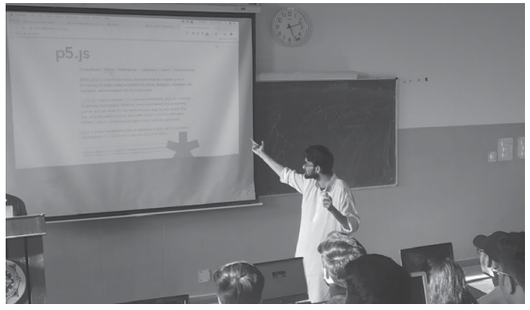



FIG 009-05 Shaharyar Shamshi conducted a workshop on p5.js for undergraduate students at the National Institute of Technology Hamirpur.
A teacher points to a projection at the front of a classroom of the p5.js website. Rows of students with open laptops look on.

Different contributors also worked on a number of documentation and education projects to strengthen and diversify the p5.js community. Qianqian Ye aims to make p5.js more accessible in China, especially within the underrepresented women and non-male identified groups. To counter the fact that most online educational resources such as YouTube are banned in China, she recorded p5.js video tutorials for beginners in Mandarin and shared them on Chinese video sites. She also partnered with female creative coders in China to host p5.js workshops for girls, women, and nonbinary-identified people, as well as post interviews with role models in the p5.js community on Chinese social media.

Manaswini Das, Nancy Chauhan, and Shaharyar Shamshi worked to empower people of the Indian community of all backgrounds to learn how to program. Through Hindi translation efforts, they were able to provide tools to the Indian community in their native language, and prepare educators by collaborating with



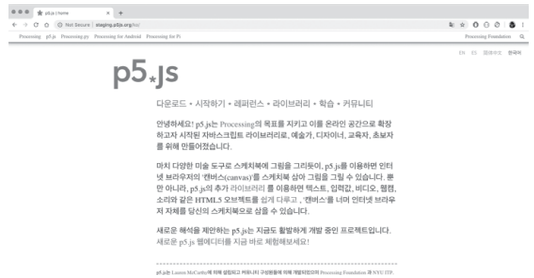



FIG 009-06 Home page for p5.js website translated to Korean.
The homepage of the p5.js website is pic-tured with a white background and a combination of black and magenta text translated to Korean.

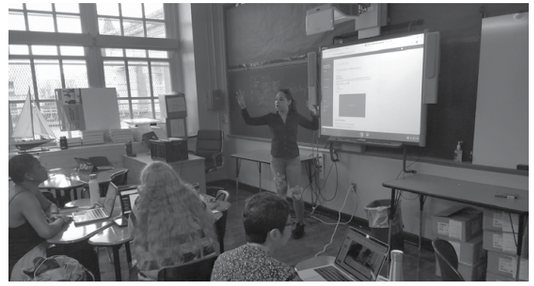


FIG 009-07 At the CS4ALL Computer Science Institute in NYC, Layla Quinones trained middle-school teachers to teach the Creative Web course, which uses p5.js to teach computational thinking to students.
Four people in a classroom. The teacher, a woman in front with dark hair, has both arms raised speaking.

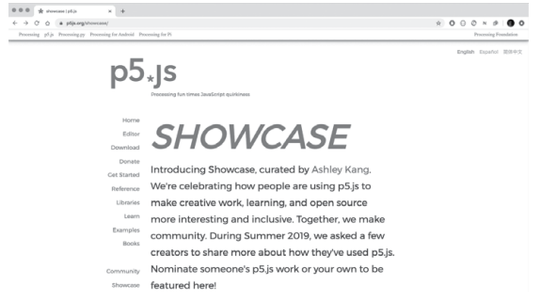



FIG 009-08 p5.js Showcase created by Ashley Kang
Screenshot of p5js.org/showcase with text introducing the project.

various NGOs and individuals to build a more diverse community around software.

Matilde Wysocki developed a p5.js curriculum and taught homeless TGNC (trans and gender nonconforming) youth basic programming as a means of self expression and digital literacy in a personal security and queer privacy context. As a peer advocate, they exposed their community to computational design and machine learning as either a means of digital literacy or personal empowerment.

Yeseul Song translated the p5.js website and reference to Korean, adding to our current set of translations including Spanish, Chinese, and Hindi (in progress).

Layla Quinones and Emily Fields, mentored by Educational Community Director Saber Khan, wrote curriculum that teaches students how to integrate sound, animation, movement, and interactivity into creative computational artifacts in p5.js. Their work focused on developing tools for urban teachers who have little experience teaching topics in CS to their own communities.

Finally, Ashley Kang created a p5.js gallery to showcase p5.js projects from around the world, with a focus on featuring artists, coders, and makers with underrepresented backgrounds. This showcase will launch along with our 1.0 release later this month.

FIG 009–09 Collection of contributor images for p5.js 1.0 Contributors Zine.
Grid with a wide range of images and colors.


FIG 009–10 Contributor Kate Hollenbach with her students.
Group of students posing in front of project p5.js website smiling and waving.

Next Steps

One of the key decisions we made at the Contributors Conference was that going forward, "p5.js will not add any new features except those that increase access." We hope that this commitment will focus the future library around our priorities of inclusion, diversity, and accessibility. We imagine this may open many conversations into the different ways we can increase access, and the range of barriers and structures that work against it. We have a large-scale vision for how the p5.js library can significantly push forward web accessibility, and we want to continue researching and prototyping that in partnership with disabled programmers and artists, students, and institutions.

We'll also be publishing our p5.js 1.0 Contributors Zine in March. This book will celebrate all the contributors to p5.js and the work we've done together. Stay tuned for an announcement about it soon!

Finally, we will be following through on the challenge we've set forth for ourselves– to open the project in a new way by implementing a rotating model of leadership! Lauren Lee McCarthy will be stepping down from her role as project lead, making space for exciting new ideas and leaders in the project. Look out for an open call for p5.js Project Lead coming later this month.

More soon! We're so looking forward to digging deeper into all this work after our 1.0 release, and engaging the community in deeper and more expansive ways. For now, we want to say biggest thanks to all of you artists, makers, teachers, students, supporters, and contributors for being a part of it. p5.js would not be what it is, and we would not be who we are, without you. <3



P5* 010 p5.js x W3C: Web We Want, 2020.

Each year, the World Wide Web Consortium (W3C), a nonprofit organization where web technologies like HTML and CSS are designed and standardized, organizes a week-long meeting of its entire community. This meeting, known as TPAC (Technical Plenary & Advisory Committee meeting), has one portion dedicated to public breakout sessions, where topics range from how to get started contributing to W3C, to privacy and security in HTML, web accessibility, WebXR, and other new technology standards on the web. For the 2020 TPAC, Lauren Lee McCarthy worked in collaboration with W3C and Bocoup to curate newbie- and artist-friendly sessions. These sessions opened wide-ranging conversations between W3C contributors, p5.js and Processing community members, and many who were participating in the W3C community directly for the first time.

Creative Imagination for an Ethical Web

Mindy Seu, Ashley Jane Lewis, shawné michaelain holloway, and Amelia Winger-Bearskin reflected on their diverse practices and how each points us toward a future vision of an ethical internet.

Mindy Seu introduced her recent cyberfeminismindex.com project, which aims to trace cyberfeminism from the '90s to today, bringing together global and intersectional perspectives, working in a way that is permeable, open-source, and crowdsourced. Mindy's comments about design and form when considering how the index would be presented, and her reference of Paul Soullelis's idea of "download as an act of protest" were especially impactful.

Ashley Jane Lewis presented her practice, which is grounded in ideas of Afrofuturism, and her thinking about speculative justice and community engagement within art and tech. In Investing in Futures: Beyond Policing, Ashley worked with a team to create a card game of creative constraints that asked participants to collaboratively imagine a world where community is based not on policing but on care. She also discussed her work with ml5.js, which is an open-source machine-learning tool and a sister project to p5.js. Within that project, she has been leading an effort to reconsider the role of codes of conduct and community agreements, asking "How do we articulate our responsibilities to the community?"

Amelia Winger-Bearskin opened by noting that when Thomas Jefferson and Benjamin Franklin were drafting the U.S. Constitution, they spent time learning from the Iroquois Confederacy. Many of the principles of the Constitution were directly based on their confederacy, but they left out ideas about the social and cultural networks that sustain the Iroquois Confederacy. Amelia points out how this "I see it, I like it, I want it, I

FIG 010-01 Amelia Winger-Bearskin presenting a video work with abstract-generated visuals. Brown and purple Bézier curves mirrored across the screen, folding inward.

take it" approach is still embedded in tech today, which she likens to building a project without checking its dependencies. As she put it, "The problem with colonial mindset is one of underfitting: extracting an idea without the context which made that idea work in the first place." She urged us not to ignore data, but to seek data and perspectives from diverse sources. "Don't colonize our future!"

Finally, shawné michaelain holloway read from her paper, "Of the Web as Homefront: In Regard to Freedoms of Speech and Freedoms In General," which urgently called for a rethinking of the web. She pushed us to become aware of our encoded expectations, saying "When [the web as we know it] became the internet standard, the white, linear, page-focused, copycat structure of the book became locked into our imagination and has since dictated what kinds of content can and should be created for the entirety of the digital space." Instead, she proposed a model of "refactoring" to rearrange the web.

The process of refactoring can feel like ripping apart a codebase down the middle and holding the raw chunks in your hands, then trying to stitch it back together in a different form. Some pieces inevitably get left behind and we discover new ones which must get made. It's a process that feels messy and exhilaratingly risky, but you take on that risk because you've recognized that the current structure is broken and cannot continue with patches alone. The feeling of a successful refactor as limitless. And as shawné says, "Such limitlessness is also what makes [the web] such a beautifully mysterious, risky, and desirable place to be."

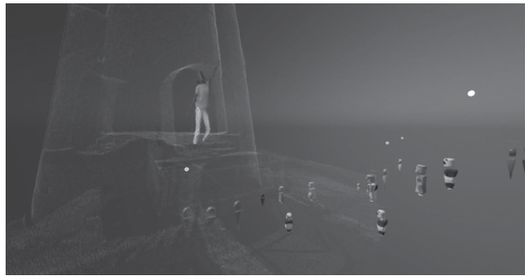


FIG 010-02 Valencia James performing using her Volumetric Performance Toolkit. 3D representation of Valencia dances on white platform structure while avatars watch.

Accessing WebXR Through Art

Valencia James, Stalgia Grigg, and Evelyn Eastmond presented their work developing platforms and tools that open access to WebXR through art, in a time when remote experimentation is desperately needed.

Evelyn Eastmond presented her work with M Eiffler, where they worked with art students to develop an XR platform that supported a remote critique space.



They collected data about the classroom experience and incubated new tools, including a prototype in Mozilla Hubs where students dropped in virtual models, videos, and sounds for discussion. They found that by using this tool, students could again find a sense of connection to their peers and art work and regain a sense of community that Zoom wasn't offering. Evelyn also questioned Microsoft's role in this space, asking if "Microsoft wants to support not just the future of the open web for developers but if it also wants to support the future of the open web for spatial and embodied making as well."

Valencia James introduced her project AI_am, which explores how artificial intelligence (AI) and contemporary dance can inform and advance each other. In a duet between AI and a human dancer, Valencia blurs the distinction between virtual and physical as well as student and teacher. This work informed her current project, the Volumetric Performance Toolkit, which is a set of tools that allows movement artists to capture and stream their performances in real-time from their own living spaces, using minimal equipment, to be viewed by a virtual audience of real people. Central to this work is creating an accessible tool that supports "artists of all ethnicities, cultures, and abilities to participate."

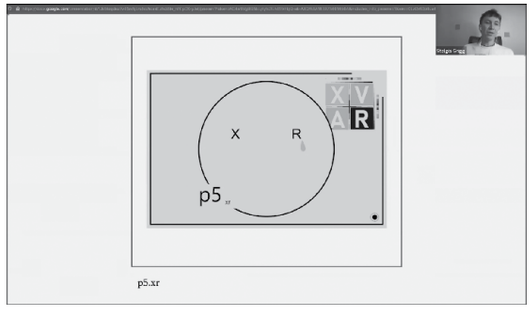



FIG 010-03 Stalgia Grigg presents their p5.xr library. Slide with yellow p5.xr screen, X and R positioned like eyes in a circle with a tear dripping from the R.

Stalgia Grigg questioned how tools articulate a worldview and center a set of values for creative expression. Looking at XR pedagogy, Stalgia contrasted the Decolonizing Augmented Reality syllabus, compiled by Jessy Escobedo and Selwa Sweidan, with the commercial tool Spark AR filters. They asked, "How can we create tools that inspire deeper forms of questioning? Tools that make people want to learn how to critique power instead of further entrench it?" Stalgia walked us through a demo of the p5.xr library they created that enables people to easily make an immersive sketch. They spoke to the need for more tools in the XR space that offer accessibility without restrictions on content, or predefined judgements of value.

Consentful Communication on the Web

How do we foster consentful communication on the web? Xin Xin, evelyn masso, and aarón montoya-moraga shared their experiences working to welcome newbies on GitHub, centering developer discussions around accessibility, considering the cultural implications of internationalization, and building open-source chat tools that

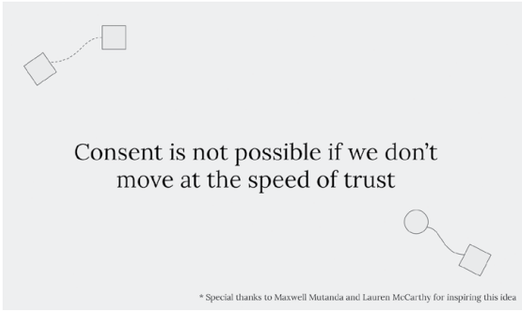



FIG 010–04 Screenshot from Xin Xin's presentation. Slide reading "Consent is not possible if we don't move at the speed of trust" with circles and squares connected via wavy lines.

think through security and memory in conversation.

evelyn masso talked about encoding values in open source. What often gets lost on GitHub is the fact that there are two or more people on each side of an interaction. How do you set boundaries for maintainers and contributors, and how do you document policies to support these boundaries? They offered the p5.js Access Statement as one example of using documentation to maintain a project's values in development. The statement makes explicit "a commitment to only add features to p5.js that increase access (meaning inclusion and/or accessibility). This means considering the vectors of diversity (e.g., gender, social, economic, race, ethnicity, language, disability, etc.) that can impact access/participation; and taking action to acknowledge, dismantle, and prevent barriers. We prioritize the needs of historically marginalized groups over the continued comfort of more privileged groups with p5.js."

In a very personal talk, aarón montoya-moraga talked about their experience as a Chilean, non-native English speaker working in open source. They shared their constant doubts: "Am I doing enough / too much? Using the right tone and words? Is it colonizing to write in English? Is it a transparent way to communicate? Do I come across as nice or condescending? How do I think through the side effects of what we do?" They pointed toward key documents that guide them, including the p5.js Community Statement, p5.js Contributor Docs, Berlin Code of Conduct, and Chatam House Rule.

Xin Xin discussed their project Togethernet, which starts with the question, "What would social media look like if it prioritized users' intentions behind every instance of online communication?" Through a deep dive into technical infrastructure, interface design, and user testing, Xin is building a communication tool that aims to transform digital rights policies into embodied practice. Key to this work is an exploration of consent, learning from wide-ranging precedents, including the FRIES framework (Freely given, Reversible, Informed, Enthusiastic, and Specific), the Fediverse, and kinky feminist sexblog The Pervocracy.

Special thanks to Dominique Hazael-Massieux, Boaz Sender, and Sheila Moussavi for their support in organizing these sessions! We look forward to future collaborations and continued work together with W3C to make the internet more inclusive and accessible.



P5* 011 p5.js Libraries, Recorded, 2021.

Core Libraries

- p5.Sound (https://p5js.org/reference/#/libraries/p5.sound) brings the Processing approach to Web Audio as an addon for p5.js. Functionality includes audio input, playback, manipulation, effects, recording, sequencing, analysis and synthesis. The library is designed to be used in tandem with p5.js. p5.Sound is built with several modules (Clock, TimelineSignal, and signal math components) from Tone.js, an interactive music framework developed by Yotam Mann. p5.Sound was created by Jason Sigal in 2014.

  Community Libraries (https://docs.google.com/spreadsheets/d/1bUzJD0cvito7Yi_S_y8KOZyLpdzcP4fL-LoALRc8QbU/edit#gid=1849576713 ) are developed, owned, and maintained by members of the p5.js community. They are independently and generously created to share. They are all open-source and include their own reference and examples. This is the list of Libraries pulled from p5js.org/libraries on October 9, 2021.
- p5.animS (https://wixette.github.io/p5.animS/) animates p5.js shapes by rendering their drawing processes. Created by Yonggang Wang.
- asciiart (https://tetoki.eu/asciiart) is a simple and easy-to-use image-to-ASCII art converter for p5js. Created by Pawel Janicki.
- p5.bezier (https://github.com/peilingjiang/p5.bezier) helps draw high-degree Bézier curves with unlimited points. Created by Peiling Jiang.
- p5.ble (https://itpnyu.github.io/p5ble-website/) enables communication between BLE devices and p5 sketches. Created by Yining Shi, Jingwen Zhu, and Tom Igoe.
- p5.bots (https://github.com/sarahgp/p5bots) lets you interact with your Arduino (or other microprocessor) from within the browser. Use sensor data to drive a sketch; use a sketch to drive LEDs, motors, and more! Created by Sarah Groff-Palermo.
- p5.button (https://www.nicoatek.com/js_libraries/p5.button.min.js) is a small, one function library to design and render simple or styled buttons directly in p5.js canvas. Created by Nicolas ATEK.
- p5.buttons (https://github.com/koerismo/p5.buttons) is a library that aims to decomplicate scripting buttons. Created by Jadon L.
- p5.clickable (https://github.com/Lartu/p5.clickable) is an event-driven, easy-to-use button library for p5.js. Created by Martín del Río.
- p5.cmyk.js (https://github.com/jtnimoy/p5.cmyk.js) is a tool for CMYK ColorSpace. Created by JT Nimoy.

- p5.collide2D (https://github.com/bmoren/p5.collide2D) provides tools for calculating collision detection for 2D geometry with p5.js. Created by Ben Moren.
- Concave Hull (https://github.com/markroland/concaveHullJS) calculates a Concave Hull from a set of points in 2D space. Created by Mark Roland.
- CO2Budget.js (https://github.com/OlafVal/CO2Budget.js) opens up the idea of the CO2-Countdown to creative communities of coders, designers and artists associated with p5.js and JavaScript. Created by Olaf Val.
- cursor.js (https://raw.githubusercontent.com/Aayush95f/cursor/main/cursor.js) helps you to create different cursors at different times with JavaScript. Created by Aayush.
- c2.js (http://c2js.org) c2.js is a JavaScript library for creative coding based on computational geometry, physics simulation, evolutionary algorithm, and other modules. Created by Ren Yuan.
- p5.Create (https://github.com/zachmohammed/P5.Create) is a library which provides save, animation, collision and tag functions designed for efficient game creation. Created by Zach Mohammed.
- p5.createloop (https://www.npmjs.com/package/p5.createloop) creates animation loops with noise and GIF exports in one line of code. Created by Peter Hayman.
- p5.dimensions (https://github.com/Smilebags/p5.dimensions.js) p5.dimensions extends p5.js's vector functions to work in any number of dimensions. Created by Smilebags and Max Segal.
- p5.EasyCam (https://github.com/freshfork/p5.EasyCam) is a simple 3D camera control with inertial pan, zoom, and rotate. Major contributions by Thomas Diewald. Created by jWilliam Dunn.
- p5.experience (https://github.com/loneboarder/p5.experience.js) adds additional event-listening functionality for creating canvas-based web applications. Created by Felix Meichelböck.
- p5.func (https://idmnyu.github.io/p5.js-func) is a p5 extension that provides new objects and utilities for function generation in the time, frequency, and spatial domains. Created by R. Luke DuBois.
- p5.geolocation (https://github.com/bmoren/p5.geolocation) provides techniques for acquiring, watching, calculating, and geofencing user locations for p5.js. Created by Ben Moren.
- p5.gibber (https://charlie-roberts.com/gibber/p5-gibber/) provides rapid music sequencing and audio synthesis capabilities. Created by Charlie Roberts.
- p5.glitch (https://p5.glitch.me/) extends p5.js for glitching images and binary files. Created by Ted Davis.



- grafica.js (https://github.com/jagracar/grafica.js) lets you add simple but highly configurable 2D plots to your p5.js sketches. Created by Javier Graciá Carpio.
- p5.gui (https://github.com/bitcraftlab/p5.gui) generates a graphical user interface for your p5.js sketches. Created by Martin Schneider.
- I_AM_UI (https://github.com/zturtledog/I_AM_UI/blob/main/I_AM_UI.js) is a UserInterface library. Created by sam oakes.
- p5.jacdac (https://microsoft.github.io/jacdac-docs/clients/p5js) enables you to plug-and-play micro-controllers for p5.js. Created by Jonathan de Halleux.
- p5.joystick (https://github.com/Vamoss/p5.joystick) enables you to connect and play with physical joysticks. Created by Vamoss.
- p5.j5 (https://github.com/monteslu/p5.j5) is a full Johnny-Five.io Nodebots/robotics library for Arduino and other microcontrollers made to work directly in browser with WebSerial or WebUSB. Created by Luis Montes.
- p5LiveMedia (https://github.com/vanevery/p5LiveMedia) is a real-time audio, video, canvas, and data sharing. Created by Shawn Van Every.
- p5.localmessage (https://github.com/bmoren/p5.localmessage) provides a simple interface to send messages locally from one sketch to another for easy multi-window sketching! Created by Ben Moren.
- marching (https://github.com/jtnimoy/marching) supports raster to vector conversion, isosurfaces. Created by JT Nimoy.
- p5.math.js (https://editor.p5js.org/bharathsatheesan7a/sketches/NHvinWyye) converts mathematicla operations of JavaScript into simple human language so that they can be understood really quickly. Created by Bharath Satheesan.
- mappa (https://github.com/cvalenzuela/Mappa) provides a set of tools for working with static maps, tile maps, and geo-data. Useful when building geolocation-based visual representations. Created by Cristóbal Valenzuela.
- ml5.js (https://ml5js.org/) builds on Tensorflow.js and provides friendly access to machine learning algorithms and models in the browser. Created by NYU ITP/IMA and contributors.
- número (https://github.com/nickmcintyre/numero) is a friendly and intuitive math library for p5.js. Created by Nick McIntyre.
- p5.particle (https://github.com/bobcgausa/cook-js) can be used to create data-driven effects which are defined through user structures or JSON input and user-draw functions. Created by Robert Cook.
- p5.party (https://github.com/jbakse/p5.party) is a library for easily creating networked multiuser games, apps, and sketches. Created by Justin Bakse.

- p5.pattern (https://github.com/SYM380/p5.pattern) is a pattern drawing library for p5.js. Created by Taichi Sayama.
- physics diagram language (https://github.com/phywand/pdl-library/) is a physics language for developing static and interactive diagrams on the web. Created by Ian Lawrence.
- p5.play (http://p5play.molleindustria.org/) provides sprites, animations, input and collision functions for games and gamelike applications. Created by Paolo Pedercini.
- p5.Polar (https://github.com/liz-peng/p5.Polar) provides mathematical abstractions making it easy to create beautiful, kaleidoscopic, radial patterns. Created by Liz Peng.
- p5.quadrille.js (https://objetos.github.io/p5.quadrille.js/) is a p5.js quadrille library. Created by Jean Pierre Charalambos.
- react-p5 (https://github.com/Gherciu/react-p5) lets you integrate p5 Sketches into your React App. Created by Gherciu Gheorghe.
- p5.recorder (https://github.com/doriclaudino/p5.recorder) records sketch canvas and audio. Created by Dori Claudino.
- p5.Riso (https://antiboredom.github.io/p5.riso) is a library for generating files suitable for Risograph printing. It helps turn your sketches into multi-color prints. Created by Sam Lavigne and Tega Brain.
- rita.js (https://rednoise.org/rita) provides a set of natural language processing objects for generative literature. Created by Daniel C. Howe.
- Rotating knobs (https://codeforartists.com/RotatingKnobMaker) makes knobs you can rotate with custom graphics and return value ranges. Created by Miles DeCoster.
- p5.scenemanager (https://github.com/mveteanu/p5.SceneManager) helps you create sketches with multiple states / scenes. Each scene is a like a sketch within the main sketch. Created by Marian Veteanu.
- p5.screenPosition (https://github.com/bohnacker/p5js-screenPosition) adds the screenX and screenY functionality from Processing to p5.js. Created by Hartmut Bohnacker.
- p5.scribble (https://github.com/generative-light/p5.scribble.js) draws 2D primitives in a sketchy look. Created by Janneck Wullschleger, based on a port of the original Processing library. Created by handy.
- p5.serial (https://github.com/vanevery/p5.serialport) enables serial communication between devices that support serial (RS-232) and p5 sketches running in the browser. Created by Shawn Van Every, Jen Kagan, and Tom Igoe.
- p5.shape.js (https://github.com/gaba5/p5.shape.js) adds more simple shapes to the p5.js framework. Created by Sebastien Lorentz.
- Shape5 (https://github.com/pfe1223/Shape5js) is a 2D primative library for elementary students who are learning to code for the first time. Created by Patrick Ester.
- simple.js (https://github.com/makeyourownalgorithmicart/simple.js/wiki) provides helper functions and defaults for young and new coders. Created by Tariq Rashid.
- p5.slides (https://github.com/GarrettMFlynn/p5.js-slides) is a presentation platform written in p5.js. Created by Garrett Flynn.
- p5snap (https://www.npmjs.com/package/p5snap) is a command-line interface that creates and saves snapshots of a p5 sketch on an interval. Created by Zach Krall.
- p5.speech (https://idmnyu.github.io/p5.js-speech/) provides simple, clear access to the Web Speech and Speech Recognition APIs, allowing for the easy creation of sketches that can talk and listen. Created by R. Luke DuBois.
- p5.start2d.js (https://github.com/eltapir/p5.start2d.js) is an extension for 2D static art using pixels, millimeters, centimeters or inches. Created by Kris HEYSE.
- p5.tiledmap (https://github.com/linux-man/p5.tiledmap) provides drawing and helper functions to include maps in your sketches. Created by Caldas Lopes.
- TiledPlay (https://github.com/andrewtacon/p5.tiledplay.js) bridges the functionality of p5.js, p5.play.js, and p5.tiled.js to easily create 2D platformer games using JavaScript. Created by Andrew Tacon.
- p5.timer (https://github.com/scottkildall/p5.timer) performs asynchronous timing functions: countdowns, one-shot timers, % time elapsed and encapsulates the millis() function, and other related calculations that often clog up your code. Created by Scott Kildall.
- p5.touchgui (https://github.com/L05/p5.touchgui) is a multi-touch and mouse GUI library for p5.js. Created by Carlos L05 Garcia.
- tramontana (https://www.tramontana.xyz/) enables use of many devices (iOS, Android, tramontana Board, ...) to create interactive environments, interactive spaces or just prototype experiences at scale and in space. Created by Pierluigi Dalla Rosa.
- TurtleGFX (https://github.com/CodeGuppyPrograms/TurtleGFX) allows you to code with Turtle Graphics in JavaScript. Great for education and creative coding. Created by Adrian.
- p5.tween (https://github.com/Milchreis/p5.tween/) enables easy tween animations for your objects and shapes. Created by Nick Müller.
- vida (https://tetoki.eu/vida) is a simple library that adds camera (or video) based motion detection and blob-tracking functionality to p5js. Created by Pawel Janicki.
- p5.voronoi (https://github.com/Dozed12/p5.voronoi) provides a set of tools to draw and utilize voronoi diagrams in your p5.js sketches. Created by Francisco Moreira.
- p5.wasm (https://github.com/limzykenneth/p5.wasm) is an addon library written in Rust and compiled to WebAssembly that implements common functions found in

p5.js. Created by Kenneth Lim.

- WebMidi.js (https://github.com/djipco/webmidi) enables you to easily send and receive MIDI messages from p5. Created by Jean-Philippe Côté.
- p5.webserial (https://github.com/gohai/p5.webserial/) is a library for interacting with Serial devices from within the browser, based on Web Serial API (available on Chrome and Edge). Created by Gottfried Haider.
- p5.xr (https://p5xr.org/#/) is a library for creating VR and AR sketches with p5. Created by Stalgia Grigg.
- p5.3D (https://github.com/FreddieRa/p5.3D) supports 3D Text and Images in WebGL. Created by Freddie Rawlins.



P5* p5.js Leadership
012 Transition, 2020.

From 2013–2020, the p5.js project was led by Lauren Lee McCarthy. Following the release of 1.0 and discussions at the 2019 p5.js Contributors Conference, p5.js began a transition to rotating leadership (https://medium.com/processing-foundation/making-space-for-the-future-of-p5-js-d3c6bd3da9ac) the next phase of p5.js:

After nearly seven years of developing and leading the p5.js project, I will be leaving my role as project lead in January 2020. It's time to open up space for others to lead, and to build new projects and communities. The dream I have for this project is to move to a model of rotating leadership. I want to see a one-year project lead fellowship position through the Processing Foundation that offers a paid opportunity to steward and lead the project in whichever direction this person chooses. Former project leaders would act as mentors for them, helping with handoffs and transition from year to year. I believe rotating leadership would best facilitate new ideas and perspectives guiding the project. It would open opportunities for more people to lead and learn from this experience in a supported and sustainable way.

I want to express my deep gratitude to the thousands of contributors, teachers, and creators. There are too many to name, but here is a partial list (https://github.com/processing/p5.js#contributors) (please add yourself if your name is missing). Many of you have been contributors over years, guiding the project to key milestones and new directions. The project will be strengthened by making space for all these new leaders. My hope is that p5.js continues and grows, and that long-time and newbie contributors alike feel welcome, supported, and empowered as we take this next step for the project.

In April 2020, the p5.js community embarked on a process to transition to a rotating leadership model. We held an open call for our inaugural rotating p5.js project lead, whose role over the one-year term is to steward the p5.js project, software, and community:

In February, we reached our 1.0 Release, a significant milestone in terms of software, documentation, and community. Now, we are seeking a motivated individual to act as a catalyst for the next phase of growth, alongside our productive and prolific community, and in service of our collective investment in exploring the creation of art and design with technology. The p5.js Project Lead's role will be to continue stewarding the p5.js project, software, and community, providing leadership in creating generative spaces to gather, speak up, and participate.

p5.js is an open-source community whose ethics transfer to our working model. Rather than using a top-down model of leadership, we are committed to inclusion and access. We value the work of the collective equally and seek a candidate that has a demonstrated interest in facilitating, guiding, collaborating, and making coding accessible and inclusive for artists, designers, educators, beginners, and anyone else! We are eager to work with someone who will lead the project with imagination, initiative, and independence. We are committed to supporting this trajectory through mentorship provided by previous p5.js Fellows and project leads. The position will be supervised by the Processing Foundation, which will also provide support and collaboration.

A group of 15 volunteers from the community guided the process through an open call, interviews, and final selection. The selection team considered each applicant's experience and potential for organizing, community building, technical direction, and commitment to diversity, inclusion, and access. We looked especially for the ability to work openly and collaboratively, considering how this role could support learning for the community as well as the lead. We asked how the p5.js community could provide collective mentorship to support the project lead stepping into this new role. It was a very difficult decision with so many visionary applicants. Moira Turner was selected as the next p5.js Project Lead, and led the project from August 2020–February 2021. She described experiences that prepared her for the role in her introductory medium post, "Always look at where you want to go – not where you don't want to be." (https://medium.com/processing-foundation/always-look-at-where-you-want-to-go-not-where-you-dont-want-to-be-69f82ba58762):

FIG 012–01 Moira Turner is a queer Black woman who studied Anthropology and African American studies at the University of Southern California. After graduation, she taught computer science to elementary and middle school students for three years. This experience teaching exposed her interest in coding and passion for inclusive programming spaces. Now she is the current project lead for p5.js and is excited to work with p5's multifaceted community.

> In motorcycling there's a concept called hyperfocus. When trying to avoid an object in the road, you should look towards your destination, not at the object you are trying to avoid. Many new riders get caught up staring at the object they want to avoid and inevitably run into it. Why? Because we steer our bodies to whatever our eyes are fixed on. Always look at where you want to go – not where you don't want to be. The deficit of representation of people of color in CS leaves students without role models, which exacerbates stereotypes and perpetuates the cycle. It's important that historically marginalized students have role models so they can imagine themselves in this field and, by doing so, actualize that reality by steering towards it.
>
> When I arrived at Emerson, I would work in collaboration with site teachers to build an environment where students who often don't feel successful can see themselves in a positive light. We worked to build inclusive experiences that allowed young, underrepresented students the opportunity to build foundational computational-thinking skills and promoted interest in a future in CS and related fields. Because historically marginalized groups are often deprived of aspirational role models in tech, relationship-building and intentional environments were key in providing my students with a destination to move towards. As Maya Angelou so aptly put it: "People will forget what you said... but people will never forget how you made them feel."

In April 2021, evelyn masso and Qianqian Ye became the next p5.js project Co-Leads. evelyn described their early experiences with p5.js in their introductory Medium post, beginnings (https://medium.com/processing-foundation/beginnings-by-evelyn-masso-7676719ba4a0):

> I got involved with p5.js itself much later (around 2016) when I started facilitating "Intro to Contributing to Open Source" workshops with Emma Cunningham and Rebecca Bever as part of Write/Speak/Code (https://www.writespeakcode.com/) Los Angeles. I felt a lot of responsibility to the folks who were contributing to an open-source project for the first time; I wanted to give them an environment of support, respect, and care. It's a vulnerable process, and the way open-source projects treat newcomers makes a huge difference. p5.js was the only project I knew I could rely on to provide that welcoming space. I'd already contributed to other open-source projects, and I had learned the basics of how p5.js worked so I could facilitate these workshops. I even fixed a couple bugs (https://github.com/processing/p5.js/pull/2127) that we ran into during the workshop.
>
> When I was a p5.js Fellow in 2019 (https://github.com/processing/p5.js/pull/2127), I came in with an engineering-heavy mindset. I kept asking myself, "What could I code for p5.js?" and then doing it. Before long, that didn't feel rewarding for me; I was treating myself like a resource to extract labor from, without regard for what felt fulfilling for me. I had to revisit why I had first come to p5. Two to three years isn't a long time, but I felt pretty far from the workshops I had initially done with p5. I started doing workshops again, leaning into conversations about what was uniquely

FIG 012–02 Qianqian a is a Chinese artist, creative technologist and educator based in Los Angeles (Gabrielino-Tongva Land). Trained as an architect, she creates digital, physical, and social spaces exploring issues around gender, immigrant, power, and technology. She received a Master of Landscape Architecture from Cornell University and was a Processing Foundation Fellow in 2019. She currently teaches creative coding at USC Media Arts + Practice, and serves as a p5.js co-lead at Processing Foundation.

Qianqian, a non-binary Chinese person with black short hair, wearing a black tank top, standing next to an apple tree, with a lush garden in the background.

important about the project and community of p5.js.

Since then, I've realized that I don't work on p5.js so I can write code, or even so I can use it to make art. I work on p5.js to build and enjoy community.

Qianqian described their approach to leadership as "Tending the p5.garden (https://medium.com/processing-foundation/tending-the-p5-garden-c4f3e666ca39)" in their own introductory post:

Growing up in Wenzhou, a coastal city surrounded by mountains in Southeast China, I spent a good amount of time in my grandparents' garden. There were patches of bok choy, sweet potatoes, and scallions planted in soda cans when my grandmother didn't have extra planters. The neighbors often came to hang out in the garden, bringing produce, like bittermelon, that they'd just picked from their yard as gifts. This inspired a childhood dream of mine, of building a community gathering space where people can meet, chat, and maybe exchange produce.

Teaching p5.js has formed most of my understanding about this project. In 2019, I made a series of p5.js video tutorials in Mandarin called Qtv (https://medium.com/processing-foundation/interview-with-2019-fellow-qianqian-ye-799c0115c295) as a Processing Foundation Fellow. The project started with the idea of teaching my mom, who lives in China and doesn't speak English, how to code with p5. I wanted to make the coding tutorials feel intimate and grounded, so I recorded most of them in my backyard, with birds chirping, leaves rustling, and an avocado tree in the background. I was interested in exploring non-Western narratives while teaching coding, so I used examples like Chinese calligraphy brush and urban villages in Shenzhen when I explained what random() is. I currently teach creative coding at USC Media Arts



FIG 012–03 evelyn masso (she/they) is a person (all the time), a tech worker (on weekdays), and a poet (on weekends). She was a p5.js Fellow in 2019 and has spoken about issues of access in software at venues like the Frank-Ratchye STUDIO for Creative Inquiry, the W3C TPAC, and Write/Speak/Code. Originally from Ohio, she currently lives on unceded Tongva land (near Los Angeles) with a rapidly growing collection of houseplants. She spends her free time rollerskating, making silly jokes, and reading queer fiction.

evelyn wearing a collared white shirt, standing in front of a stone wall covered in ivy. She is light skinned with shoulder length brown hair and looking at the camera, smiling slightly.

+ Practice, where I introduce the students to coding through p5. I've learned a lot from my students about how accessible and inclusive p5 is. I have seen how it makes them feel less scared about programming, and how quickly it empowers students from all over just as it did for me. It makes them feel seen, welcomed, and included. [...]

Leading p5 is like tending a garden, similar to my grandparents' vegetable garden and the botanical garden at CMU. The p5 garden is the warm gathering space my younger self imagined, where people are welcomed to meet, chat, learn, and exchange ideas. I'd love to invite more gardeners to join the p5.garden, no matter how much gardening knowledge they have right now. Bring your seeds, let's plant together. If you don't have a planter, bring whatever you might have, like the used soda cans?

COMMUNITY DAY (COM) 001–006 265

Processing Community Day (PCD) celebrates art and code. PCD aims to make learning how to program and learning how to make creative work with code accessible to diverse communities, especially those who have not had access to these tools and resources. The first PCD was in Cambridge, Massachusetts in 2017 and the second was in Los Angeles in 2019. Also in 2019, we broadened the reach and impact of the PCD community by encouraging the organization of hundreds of PCDs around the world. Over 100 events were organized on six continents. PCD is now an annual, worldwide event that has operated mostly online through the pandemic. PCD 2021 celebrated the 20th anniversary of Processing.

COM 001

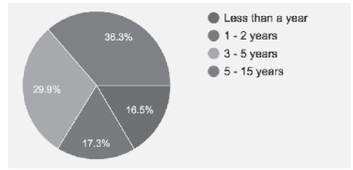
## 2016 Community Survey, 2016.

We developed the Community Survey to better understand how and why our software is used, and to get a better grasp on the communities of people who use the software. From high schools, to universities, to the home, how are people working with our software? Across the globe, who is using our software? What is working and what can be improved?

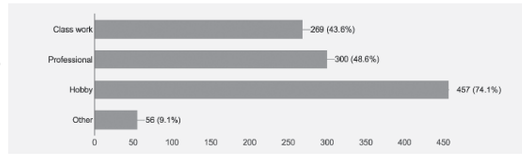
We announced the Survey through a mass email to our existing list and through Twitter and Facebook. We received over 600 responses from the middle of November to the end of the year. We're extremely grateful for the hundreds of thoughtful responses and have summarized them here.

### General Information

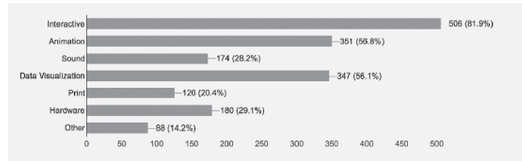
For the survey, we wanted to learn which of our software platforms people are using and why. Are people using only Processing (https://processing.org/), p5.js (https://p5js.org/), or are they using both? What percentage is using Processing for Android (https://android.processing.org/) or Processing.py (https://py.processing.org/)? Do we have people who have used our software since it was first released fifteen years ago, or are most people new? Also, we were curious about what the software is used for and if people are using it to generate income and/or for enjoyment?



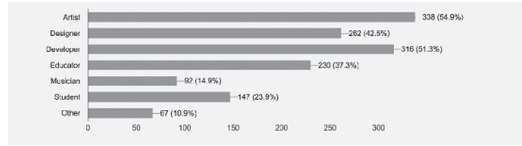

How long have you been using our software? (619 responses)
Pie chart: 36% less than one year, 30% 1-2 years, 17% 3-5 years, 17% 5-15 years




How do you use our software? (617 responses)
Bar chart: 43% classwork, 41% professional, 74% hobby, 9% other

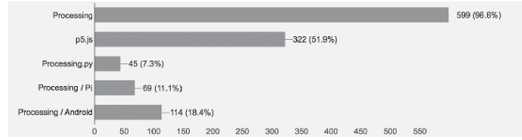



What are your main interests? (618 responses)
Bar chart: 82% interactive, 58% animation, 26% sound, 56% data viz, 20% print, 29% hardware, 14% other

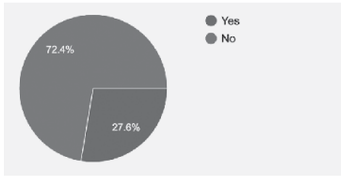



How do you identify? (161 responses)
Bar chart: 55% artist, 43% designer, 51% developer, 37% educator, 15% musician, 24% student, 11% other


How often do you use our software? (620 responses)
Pie chart: 38% once a week, 26% varies, 21% every now and then, 11% every day, 5% once a month




Which tools from the Processing family do you use? (620 responses)
Bar chart: 97% Processing, 52% p5.js, 7% Processing.py, 11% Processing for Pi, 18% Processing Android




Do you use our software to generate income? (619 responses)
Pie chart: 72% yes, 28% no

### Education

The original Processing software started as a platform for teaching and learning as well as a code sketchbook. Like Processing, the p5.js and Processing.py projects have a strong focus on learning to code. For the survey, we hoped to learn more about who is teaching with our software and in which kinds of classes, both the grade level and subject.

### International Communities

One of the most fascinating and essential aspects of our software is the international scope of the communities. For this survey, we received survey results from 68 countries, from each continent except Antarctica. Algeria, Argentina, Australia, Austria, Belgium, Brazil, Bulgaria, Canada, Chile, China, Colombia, Croatia, Czech Republic, Denmark, Dominican Republic,

Egypt, El Salvador, England, Estonia, Finland, France, Germany, Ghana, Greece, Hong Kong, Hungary, Iceland, India, Indonesia, Iraq, Ireland, Italy, Japan, South Korea, Kuwait, Lithuania, London, Luxembourg, Maldives, Mexico, Netherlands, New Zealand, Nigeria, Norway, Pakistan, Paris, Peru, Philippines, Poland, Portugal, Quebec, Reunion, Romania, Russia, Scotland, Serbia, Singapore, Slovakia, Slovenia, Spain, Sweden, Switzerland, Taiwan, Turkey, Ukraine, United States, Uruguay, Wales.

Four people sitting on the floor talking with a group of people standing behind them.

Presenters and Facilitators:

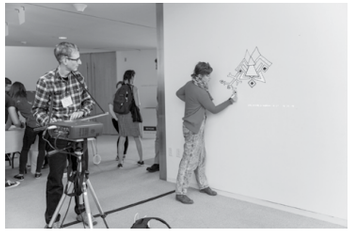
Eva Díaz, Sydette Harry, Mark Schifferli, Olivia Glennon, Johanna Hedva, Ben Fry, Casey Reas, Lauren McCarthy, Daniel Shiffman, Chris Coleman, Golan Levin, Tega Brain, Jose Luis Garcia del Castillo, Cassie Tarakajian, Gottfried Haider, Jakub Valtar, Andrés Colubri, Sharon Lee De La Cruz, Aarón Montoya-Moraga and Guillermo Montecinos, Claire Kearney-Volpe and Mathura M Govindarajan, Rosa Weinberg, Maxime Damecour, Brad Tober, Alex Norton, Jose Luis Garcia del Castillo, Ari Melenciano, Luisa Pereira, R. Luke DuBois Jonathan Feinberg, Cristobal Valenzuela.

COM 002 Boston, 2017.

The very first Processing Community Day (PCD) was organized by Taeyoon Choi and the Processing Foundation in 2017. Taking place at the MIT Media Lab, PCD 2017 drew community members from all over the East Coast to get together to meet and share what they're working on, and to learn and collaborate in person.

A group of roughly 100 people smiling at the camera.


A group of people sitting around a table eating lunch. There's a handmade sign next to the table in the shape of a cloud that reads "focus groups."



A person is drawing art on the wall with another person controlling the projection with a keyboard.

A group of people eating lunch in the sun.

Two people gathered around a laptop amid a crowd.

COM 003 Los Angeles, 2019.

PCD @ Los Angeles was an inclusive event that brought together people of all ages to celebrate and explore art, code, and activism. The day-long



An artwork made during a Racial Pedagogy workshop called "Monsters, Synths, and Future Gods" by Angela "Mictlanxochitl" Anderson Guerrero and Jason Wyman.

An artwork consisting of different kinds of plant and flower materials arranged in a mandala-formation around three lit Catholic candles.

One of Color Coded's founding members Chris, presenting.

A person holds a microphone in front of a computer and speaks, gesticulating with their other hand. Behind them, out of focus, is part of another person's face.

event, organized by Xin Xin, featured four themed-tracks – Accessibility, Disability, and Care, Radical Pedagogy, Under the Silicon, the Beach!, and Epic Play!. Each themed track contained lightning talks and sessions presented by conference guests we invited through an open call.

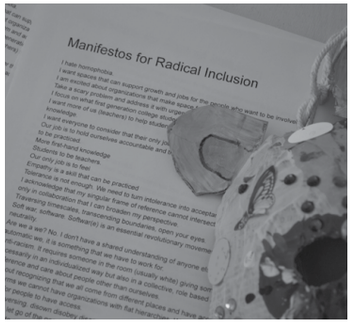
In addition to a full day of programming, we made space for anyone to share ideas and projects with the community. We set up Show & Tell Stations, a Processing Community Cafe, and a Community Open Mic Session for participants to sign up on the day of the event. The program wrapped up with an after-party consisting of performances, food, and drinks.

Speakers and Session Organizers

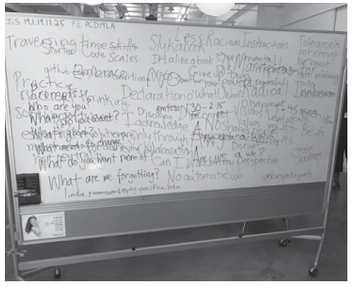
Angela 'Mictlanxochitl' Anderson Guerrero, Patrick Michael Ballard, The Best Friends Learning Gang, Color Coded, Caitlin Conlen & Paisley Smith, Saskia Freeke, Stalgia Grigg, Cynthia X. Hua, Alden Rivendale Jones, Claire Kearney-Volpe, Luke Fischbeck, Adelle Lin, Kristin McWharter, Evelyn Masso, An Xiao Mina, Jon-Kyle Mohr, Luis Morales-Navarro, Molly Morin, Romi Ron Morrison [Elegant Collisions], Peter Polack, Linda Ravenswood, README, Emily Saltz, Jeffrey Alan Scudder, Rachel Simanjuntak, Eddo Stern, Echo Theohar, Lee Tusman, roopa vasudevan, Lark VCR, Jason Wyman.

ALIEN BODIES workshop with The Best Friends Learning Gang & Johanna Hedva.

Group of people talk around table with physical prototyping materials.




A stack of papers on a desk titled "Manifestos for Radical Inclusion." Some of the text can be seen. It reads, "I hate homophobia. I want spaces that can support growth and jobs for the people who want to be involved..." More first-hand knowledge. Students to be teachers. our only job is to feel Empathy is a skill that can be practiced....." A papier-mâché sculpture with brightly colored paint, stickers, and sequins is sitting on top of the pages.



A whiteboard covered with writing in different color inks, most of which is overlapping each other. The prompts for the workshop are in black ink and the responses are on top of it in red ink. Most of the phrases in red are included in the finished manifesto.

Organizing Team

Xin Xin, Dorothy R. Santos, Johanna Hedva, Taeyoon Choi, A.M. Darke, Tega Brain, Sam Lavigne, Chandler McWilliams, Yuehao Jiang, Reneé Reizman, Tristan Espinoza, Jules Kris, Tyler Yin, Hye Min Cho, Kuan-Ju Wu, Ariel Hahn, Christine Meinders, Tuangkamol Thongborisute, Nina Cragg.

COM 004 Worldwide, 2019–2020.

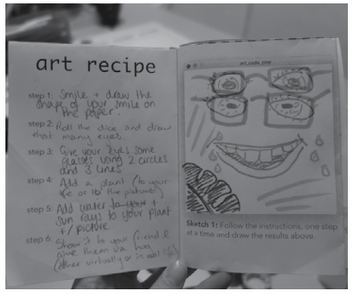
The Processing Community Day (PCD) initiative is evolving. Moving forward from its first iteration in 2017, we focused on PCD Worldwide, by supporting organizers across the world to organize PCDs in their cities. For 2020, we offered a mentorship program for PCD Worldwide Organizers who were interested in learning from past community organizers and mentors.

The goal was to help a more diverse group of organizers launch a PCD in their local communities. For PCD 2020, our mentorship program uplifted a group of advisors, created documentation, and facilitated online space to share wisdom and experiences.

A person giving a presentation at a podium.

A group of participants at PCD São Paulo smiling at the camera.


Speaker Usha Mohanraj is about to start her presentation titled, “Anyone can code - Learning to code at 50.”




One of the algorithmic zines from Processing Community Day 2019 in Bristol.

Organizers

Xin Xin, Saber Khan.

Advisors

Naoto Hieda, Becca Rose Glowacki, Aarón Montoya-Moraga, Mathura Govindarajan, Nicolas Troncosco, Qianqian Ye, Lee Tusman.

A child creating a sketch with the p5 Editor. Next to the laptop, there is a paper with potato prints of similar design.


A flyer for PCD Aarhus 2021. The flyer reads “Processing Community Day Aarhus” with the date and time of the event. The background is a multi-colored noise field.

Processing Community Day 2019, January 15–February 15, 2019.

- Aarhus, Denmark: Nathalia Novais, Tobias Stenberg Christensen, Winnie Soon & Anders Visti.
- Abu Dhabi, United Arab Emirates: Sarah Fay Krom, Craig Protzel, Ali Al-Saqban.
- Amsterdam: Creative Coding Amsterdam (Sabrina Verhage & Lisa Rombout).
- Ann Arbor, MI: Christopher Becker, Anıl Çamcı.
- Austin, TX: Natalie Freed, R.A. Robertson.
- Baltimore, MD: Andrew Bernstein, Phong Le.
- Bangalore, India: Mathura Govindarajan, Rushali Paratey, Karthik Dondeti.
- Bari, Italy: Codice Inutile.
- Basel, Switzerland: Max, Andrea, Yann, Pedro, Ted, Philipp.
- Belgrade, Serbia: Vlada, Bojan, Nemanja.
- Brisbane, Australia: David Harris.
- Bristol, UK: Becca Rose Glowacki.
- Buenos Aires, Argentina: Cristian Reynaga.
- Chicago, IL: Tiffany Funk, Allan Berry, Jon Chambers, Shawné Michaelain Holloway, Kate Hollenbach.
- Copenhagen, Denmark: Rune Madsen and Andreas Refsgaard.
- Dakar, Senegal: Marion Louisgrand Sylla.

About 60 participants of Processing Bangalore India posing for a group photograph at the end of the event. They are also holding up an interactive art piece based on data that they made through the day.


The logo for PCD Ethiopia 2019. “PCD Ethiopia” is enclosed in a hand-drawn, black circle.

- Debre Zeit, Ethiopia: Abel Wondafrash.
- Denver: Chris Coleman.
- Dublin, Ireland: Cliona Harmey.
- Dublin, Ireland: Sinead McDonald.
- Gateshead, UK: Dani Walsh.
- Geneva, Switzerland: Laurent Novac, Camille de Dieu.
- Girona, Catalonia, Spain: Joan León.
- Hong Kong: Daniel Howe, Jane Pong.
- Istanbul, Turkey: Resat Bozkir.
- Jalandhar, Punjab, India: Akash Roy.
- Kampala, Uganda, East Africa: Ezabo Baron.
- Kutztown, PA: Miles DeCoster, Josh Miller.
- Le Port, Réunion Island: Rich Porcher.
- Liège, Belgium: 3kd.be (Greg Berger) - FIG.
- Lisbon, Portugal: Marco Heleno, David Sousa-Rodrigues.
- London, UK: Danilo Di Cuia, Joel Gethin Lewis, Kenneth Lim and Tom Lynch.
- Los Angeles, CA: Xin Xin, Dorothy Santos, Johanna Hedva, Taeyoon Choi, A.M. Darke, Tega Brain, Sam Lavigne, Chandler McWilliams, Casey Reas, Lauren McCarthy, Daniel Shiffman.
- Luxembourg: Eran Goldman-Malka & Henri A.
- Madrid, Spain: Adolfo Nadal Serrano, Javier Fuentes Quijano.
- Madrid, Spain: Alfredo Calosci.
- Malmo, Sweden: David Cuartielles / Malmo University / Arduino.
- Melbourne, Australia: Mel Huang / Media Lab Melbourne.
- Mexico City: Proyectil Mx.
- Mexico City: Centro de Cultura Digital / Enrique García Alcalá / Antonio Salinas Hernández.
- Middlesbrough, UK: Jack Laidlaw.
- Milan, Italy: Codice Inutile.
- Minneapolis, MN: Ben Moren.
- Montevideo: Luisa Pereira, Brian Mackern.
- Montréal, Canada: Rodrigo Velasco + Guillaume Pelletier-Auger.
- Morelia, Mexico: Fernando García García.
- Mumbai, India: Behram, Rituparna Matkar.
- Nantes, France: Stereolux + Bérenger Recoules (L’École de design).
- New Delhi, India: Rintu Kutum, Natasha Singh and Mike Cj.
- New York, NY: Lee Tusman and Matilda Wysocki.
- Orlando, FL: Ginger Leigh.
- Ottawa, Canada: Artengine (Jason Cobill & Anthony Scavarelli).
- Pittsburgh, PA: Golan Levin, Tom Hughes (Frank-Ratchye STUDIO for Creative Inquiry / Carnegie Mellon University), Lindsey French and Kate Joranson, University of Pittsburgh.

A group of about 100 participants at PCD NYC smiling at the camera.

A group of about 40 participants at PCD Quito smiling at the camera.

- Playa del Carmen, Mexico: Erick Rivera.
- Port-Louis, Mauritius: Abdur-Rahmaan Janhangeer.
- Portland, OR: Megan McKissack.
- Porto, Portugal: Pedro Amado.
- Quito, Ecuador: Gabriel Andrade.


Three lime-green ovals are stacked on top of each other. “Processing Community Day Porto” is enclosed by these ovals.

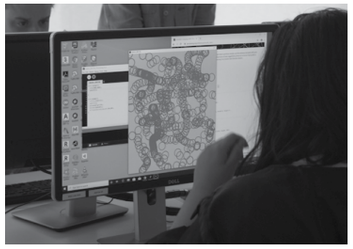
- Richmond, BC, Canada: Dethe Elza.
- Rio de Janeiro, Brazil: Marlus Araujo, Barbara Castro & João Bonelli.


A grey banner that reads “Processing Community Day Rio de Janeiro.”

- Rome, Italy: Codice Inutile.
- San Marino, Republic of San Marino: Daniele Tabellini.
- Santa Maria, RS, Brazil: Bruno Ruchiga.
- Santiago, Chile: Felicidad.

Someone placing welcome materials for PCD Santiago de Chile on a table.


A navy banner that reads “Processing Community Day - Brasil.”



A participant at PCD São Paulo on a computer. They are working on a Processing sketch of purple circles scattered across the canvas.

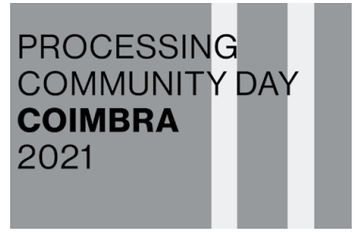
- São Paulo / Brazil: Alexandre Villares & Monica Rizzolli.
- Seattle, WA: Kai Curtis.
- Seoul, Republic of Korea: Woosung Jung.
- Seoul, Republic of Korea: W&T Lab + Seoul Express Tech Lab.
- Shanghai, China: Qianqian Ye, Chang Liu, Yuli Cai.
- Sharjah, UAE: Salem Al-Qassimi, Eleonora Cervellera, Ali Al-Saqban.
- Sheffield, UK: Richard Mather, Callum Shutt, Alex Gore + colleagues and students at Sheffield Hallam University.
- Singapore: Ong Kian Peng + Andreas Schlegel.
- Stockholm, Sweden: Jonas Johansson, Nordic Audiovisual Artists.
- Strasbourg, France: Jérôme Mutterer.
- Tampa, FL: Mikhail Mansion, Kuan Ju Wu.
- Tokyo, Japan: Naoto Hieda, Ayumu Nagamatsu & Yasuto Nakanishi.
- Toronto, Canada: Adam Herst (InterAccess member).
- Troy, NY: Nick Marchese.
- Tuzla, Bosnia and Herzegovina: Art&Sci Association “Laboratorium” Tuzla.
- Utrecht, Netherlands: Creative Coding Utrecht (Fabian van Sluijs & Saskia Freeke).
- Walla Walla, WA: Justin Lincoln.
- Wellington, New Zealand: Birgit Bachler / Creative Media Production.



Processing Community Day 2020.
January 1–December 31, 2020.

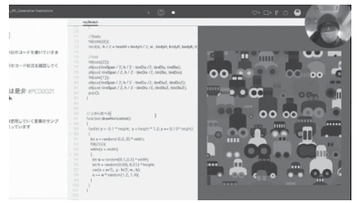
- Aarhus, Denmark: Winnie Soon and Anders Visti (DOKK1).
- Accra and Koforidua, Ghana: Fidom Hub.
- Ann Arbor, MI: Christopher Becker, Anıl Çamcı.
- Bangalore, India: Rushali Paratey, Karthik Dondeti, Mathura Govindarajan.
- Basel, Switzerland: basel.codes.
- Belgrade, Serbia: Nemanja.
- Bournemouth, UK: Edward Ward.
- Brisbane, Australia: David Harris (Southbank campus, Queensland College of Art, Griffith University).
- Bristol, UK: Becca Rose Glowacki.
- Chandigarh, India: Samarth Gulati.
- Chicago, IL: Jon Chambers.
- Ciudad Autónoma de Buenos Aires, Argentina: David Vinazza.

A child dancing in front of a projection of a slide that reads “Processing Community Day” in the top left with colorful beams coming out from the center.




A white and orange striped banner that reads “Processing Community Day Coimbra 2021.”

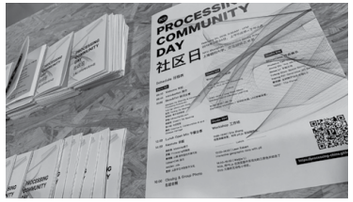
- Coimbra, Portugal: João Couceiro e Castro, João Miguel Cunha, Sérgio Rebelo, Tiago Martins, Penousal Machado.
- Copenhagen, Denmark: Rune Madsen and Andreas Refsgaard.
- Coventry, UK: Ashley James Brown.
- Coyoacan, Mexico: María Elena León.
- Debre Zeit, Ethiopia: Abel Wondafrash (Foka STEM Center).
- Guangzhou, China: Topic Society.
- Ivrea, Italy: Chiarotto Alessandro.



A screenshot of someone sharing their screen displaying a code editor and a sketch.

- Kanan, Japan: Kei Nakano (Chikatsu Asuka 近つ飛鳥).
- Kutztown, PA: Prof. Miles DeCoster, Prof. Josh Miller.
- Lisbon, Portugal: Marco Heleno, Vera Lopes.
- London, UK: UAL: Creative Computing Institute.
- Los Angeles, CA: Stalgia Grigg.
- Lugbe, Abuja, Nigeria: Michael Nwaohiri (FHA).
- Madrid, Spain: Alfredo Calosci.
- Mexico City: Steam Lab CENTRO.
- Minneapolis, MN: Ben Moren.
- Nanjing, China: Maggie Ma.
- New Delhi, India: Natasha Singh, Mike Cj, Rintu Kutum.
- New York, NY: Lee Tusman, Matilda Wysocki, rebecca marks-leopold, Todd Anderson, Oriana Medlicott, Bomani McClendon.
- Petersburg, FL: OK! Transmit.
- Portland, OR, USA: Megan McKissack.
- Porto, Portugal: Pedro Amado (Faculdade de Belas Artes da Universidade do Porto).
- Pune, India: Ritvi Mishra.
- Quito, Ecuador: Gabriel Andrade.
- Richmond, BC, Canada: Dethe Elza (Richmond Public Library).
- Rio de Janeiro, Brazil: Bárbara Castro, Carlos de Oliveira, Marlus Araujo, João Bonelli.
- San Marino: Daniele Tabellini.
- São Paulo, Brazil: Monica Rizzolli, Alexandre Villares, Cristian Cesar Cunha, João Antonio de F. P. e

Ferreira, Marco Macarena, Mariana Leal, Tatyana Zabanova.




Brochures and posters for PCD Shanghai laid out on a table.

- Shanghai and Nanjing, China: Paul Cheng.
- Stockholm, Sweden: Jonas Johansson.
- Strasbourg, France: Jerome Mutterer.
- Sydney, Australia: Luke Hespanhol.
- Tehran, Iran: Filoger.
- Tokyo, Japan: PCD Tokyo Organizers.
- Toronto, Canada: InterAccess.
- Toronto, Canada: Erin Kjaer, Tess Sutherland, Emily Green, Steve Daniels (New Media BFA program at Ryerson University).

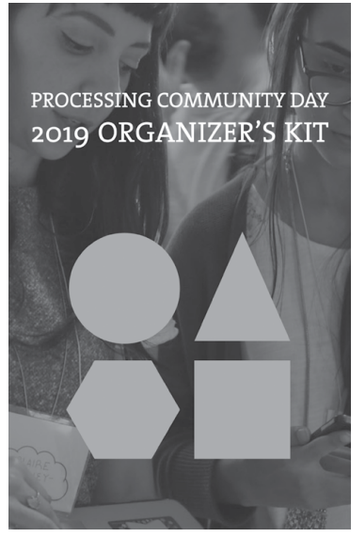
COM 005 Worldwide, Processing's 20th Birthday, 2021.

Processing was officially launched with "Release 0001 – built 9 August 2001 in Japan, 6:01:26pm (5am Eastern)." To celebrate, we organized a distributed, worldwide party! The community celebrated the 20th anniversary of Processing on August 20–22 by sharing events, zines, cakes, and artwork. #pcd2021

Community events

- Year #20 (https://openprocessing.org/curation/70077) from Openprocessing: "OpenProcessing is running a coding challenge during the month of August to celebrate the year #20 of Processing with sketches that experiment with their new dynamic logo. Submit your own sketches!"
- Dan Shiffman held a Coding Train Special: Coding with Processing Release 0001 Alpha -- Happy 20th! (https://www.youtube.com/watch?v=F8jwQP7wr6k).
- PCD 2021 Latinoamérica (https://sutidiesaboutpublication.notion.site/sutidiesaboutpublication/PCD-2021-Latinoam-rica-6e70514088ba4b259d839b611d12687f) was organized by Nicolás Troncosco.
- Processing 20th Anniversary - Denver (https://docs.google.com/forms/d/e/1FAIpQLScvVHfHUkqlnxY39lwxcpRT8or2V_jFYoBQSDqcYqXFLwBt2Q/viewform) was organized by Chris Coleman.
- Processing Community Day 2021 (https://timrodenbroeker.de/pcd2021/) was organized by Tim Rodenbroeker.
- Processing Community Hangout Japan #06 (https://pchj06.peatix.com/) was organized by Takawo Shunsuke.
- PCD Seoul (https://linktr.ee/pcd.seoul) at KOTE Space Seoul & Gather Town was organized by pcd.seoul.
- Creative Code Jam (https://www.meetup.com/creativeCodeBerlin/events/rpmjksycclbcc/) [ONLINE EDITION] was organized by Creative Code Berlin.
- Processing Community Day India 2021 (https://docs.google.com/forms/d/e/1FAIpQLSfulKyRCXBCDrLKXSKRJeixe3tf7qyGemft1IlvKlifhHsrYQ/viewform/) was organized by Processing India (Rasagy Sharma & Anushka Trivedi).
- Creative Coding Fest (https://ccfest.rocks/) was organized by Saber Khan
- Museum of Crypto Art and VerticalCrypto Art hosted an exhibition of artworks created with Processing and a panel discussion on the future of generative art with Casey Reas, Qianqian Ye, Dmitri Cherniak, and ix shells.
- Processing Community Day Copenhagen (https://pcdcph.com/) at the Royal Danish Academy was organized by Andreas Refsgaard, Stig Møller Hansen, and Rune Madsen.
- PCD Aarhus (https://pcdaarhus.net/) was organized by Winnie Soon and Anders Visti.
- PCD Taiwan (https://clab.org.tw/en/events/pcd-taiwan/) was organized by Taiwan Contemporary Culture Lab (C-LAB).
- PCD Venice Beach (http://www.brightmoments.io) was organized by Seth Goldstein.
- PCD Coimbra (http://pcdcoimbra.dei.uc.pt/2021/) was organized by Sérgio Rebelo.
- PCD Ciudad Autónoma de Buenos Aires, Argentina was organized by David Vinazza.
- PCD Latinamerica was organized by Nicolás Troncoso López.
- Ben Fry and Casey Reas were interviewed on Art Blocks.
- Raphaël de Courville did a broadcasting about Processing and Hic et Nunc on his Twitch channel.
- Computational Mama did a special Processing Community Day Stream on her Twitch channel as part of PCD India.
- Zimaldo Melo led a multiday art and code workshop (http://www.bacteria.com.br) online.
- Chelly Jin launched an updated version of Diversity with Code + Art (http://diversity.p5js.org/).
- A #sketch4processing fundraiser was organized on Hic Et Nunc by Melissa Rodriguez, Antonius Oki, p1xelfool, VerticalCrypto Art, Raphaël de Courville, and James Powderly.

COM 006 Processing Community Day Organizer's Kit, 2019, Xin Xin.




Two women gathered around a computer at Processing Community Day 2017.

Hello!

Thank you for registering to organize Processing Community Day. We understand that it can take enormous physical and emotional effort to organize a community event, therefore we have compiled this Organizer's Kit to help you get things started. The kit is licensed under a Creative Commons Attribution 4.0 International License. Special thanks to Lauren McCarthy, Casey Reas, Dorothy Santos, Johanna Hedva, and Daniel Shiffman for their guidance and support.


Luisa Pereira and Yeseul Song help facilitate a sign-language–based p5.js workshop led by Taeyoon Choi.

A Letter to the Organizers

Dear Processing Community Day Organizers,

Processing Community Day (PCD) is a day picked between January 15 and February 15, 2019 to celebrate art, code, and diversity around the world. Although we are providing a step-by-step guide to lead a PCD in this handout, keep in mind that as PCD organizers, you are welcomed to define the format of your event in any way that serves your local communities.

Asking the following questions might help define the event format –

- Which communities are you hoping to engage? Are they affiliated with an institution or do they live outside of an institution?
- What are some ways to make your event more accessible for folks who might otherwise feel excluded from the event?
- What are some ways to create a friendly, inviting environment for beginners and folks who are new to the community?
- Is your event serving the interest of your communities? How might your attendees benefit from your event?

Aside from the conventional lecture / demo format, your PCD can be a show & tell session, a zine-making session, an audio-visual performance, a DIY art show, a game jam, or even a group discussion on software art.

We are excited to learn about new ways to organize a Processing Community Day from you. Thank you for joining us for the ride!

Xin, PCD Director & Lead Organizer


People hanging out at Processing Community Day 2017.

Code of Conduct

All participants in Processing Community Day (PCD)—including, but not limited to the PCD online forum—are required to agree to the following Code of Conduct. This includes all attendees, speakers, performers, workshop leaders, patrons (sponsors), volunteers, and staff. The PCD team is dedicated to providing a harassment-

free experience for everyone, regardless of gender, gender identity and expression, sexual orientation, disability, mental illness, neurotype, physical appearance, body, age, race, ethnicity, nationality, language, or religion. We do not tolerate harassment of participants in any form. Anyone participant who violates this Code of Conduct may be limited or expelled from PCD at the discretion of the PCD organizers. This Code of Conduct is subject to change and expansion.

Be mindful of your language – Engage with community members, mentors, and contributors with respect and good intention.

Use your best judgment – If it will possibly make others uncomfortable, do not say it or post to the community forum.

Be respectful – While disagreements may arise, it is not an opportunity to attack or threaten someone else's thoughts and/or opinions. Remember to approach every situation with patience, understanding, and great care.

Be intentional – Consider how your contribution will affect others in the community.

Be open minded – Embrace new people and new ideas. Our community is continually evolving and we welcome positive change.

Definitions of Harassment include:

- Offensive comments related to gender, gender iden tity and expression, sexual orientation, disability, mental illness, neurotype, physical appearance, body, age, race, ethnicity, nationality, language, or religion
- Unwelcome comments regarding a person's lifestyle choices and practices, including those related to food, health, parenting, drugs, and employment
- Deliberate misgendering or use of 'dead' or rejected names
- Gratuitous or off-topic sexual images or behaviour in spaces where they're not appropriate
- Physical contact and simulated physical contact (eg, textual descriptions like "hug" or "backrub") without consent or after a request to stop
- Threats of violence
- Incitement of violence towards any individual, including encouraging a person to commit suicide or to engage in self-harm
- Deliberate intimidation
- Stalking or following
- Harassing photography or recording, including logging online activity for harassment purposes
- Sustained disruption of discussion
- Unwelcome sexual attention
- Pattern of inappropriate social contact, such as requesting/assuming inappropriate levels of intimacy with others
- Continued one-on-one communication after requests to cease
- Deliberate "outing" of any aspect of a person's identity without their consent except as necessary to protect other PCD community members or other vulnerable people from intentional abuse
- Publication of non-harassing private communication without consent by the involved parties

PCD prioritizes marginalized people's safety over privileged people's comfort.

PCD reserves the right not to act on complaints regarding:

- 'Reverse' -isms, including 'reverse racism,' 'reverse sexism,' and 'cisphobia'
- Reasonable communication of boundaries, such as "leave me alone," "go away," or "I'm not discussing this with you."
- Communicating in a 'tone' you don't find congenial
- Criticizing racist, sexist, cissexist, or otherwise oppressive behavior or assumptions

Consequences, Reporting and Enforcement on Processing Community Day

Please contact Xin Xin, Dorothy Santos, Johanna Hedva, Lauren McCarthy, Casey Reas, or Dan Shiffman listed on day.processing.org/people. This team of organizers will be trained in how to address and report any incidents attendees bring to our attention. You may also contact us via email at day@processing.org.

Participants asked to stop any harassing behavior are expected to comply immediately. If a participant engages in harassing behavior, the PCD team may take action we deem appropriate, up to and including expulsion from all PCD spaces (both on and offline) and identification of the participant as a harasser to other PCD community members and/or the general public.

Acknowledgements

This Code of Conduct was compiled by Dorothy Santos, based on resources provided by Geek Feminism, and borrows heavily from open-source policies authored by p5.js and XOXO Festival.

Thank you also to Lauren McCarthy, Johanna Hedva, and Xin Xin for their advice and support.





Step-by-Step Guide

Visit discourse.processing.org. Introduce yourself on the PCD @ Worldwide thread. Create a discourse thread for your city. Invite collaborators to join the discussion. Schedule an IRL planning meetup.

Find a venue

We recommend collaborating with public facilities such as libraries, schools, or community-driven spaces with good, stable WiFi such as community centers, art / culture venues, hackerspaces, cafes, tea houses, or bars.

Pay attention to the infrastructure and amenities of the space: Does the venue have fast internet connection for your group size? Does it come with equipment such as a monitor or projector? Does it have a gender-neutral bathroom? Is it child-friendly? Is the space accessible for disabled folks? We recommend this checklist to ensure that you have an accessible space.

Secure a date & time to host PCD

Find a date that works for your organizing team and the venue. Double check the calendar to make sure that it doesn't clash with other popular events in your communities.

Organize your program

Are you looking to organize a simple afternoon event or are you hoping to go further? We are currently unable to provide financial assistance for the node events. If you are hoping to organize a larger event, are there any organizations, institutions, or companies you might seek support from? How might receiving fundings from a company affect the content of your event? Can you use an organization's help to create a more accessible and inclusive space?

Here are two PCD programming templates:

4-hour Processing Community Day

- Welcome & introduction (30 min)
- Beginner-friendly workshop pt. 1 (90 min) Short break (15 min)
- Beginner-friendly workshop pt. 2 (30 min) Watch PCD @ LA lightning talks (30 min) Relaxed group discussion over refreshment (30 min) Wrap-up: discuss ways for the community to go forward and group picture! (15 min)

8-hour Processing Community Day

- Welcome & introduction (30 min)
- Community show & tell (60 min)
- Beginner-friendly workshop I (120 min)
- Lunch break (60 min)
- Guest lecture or Daniel Shiffman's Coding Train Videos (30 min)
- Beginner-friendly workshop II pt. 1 (90 min) Short break (15 min)
- Beginner-friendly workshop II pt. 2 (30 min) Community discussion on topics related to art, code, and diversity. See PCD @ LA tracks for possible topics. (45 min)
- Wrap-up: discuss ways for the community to go forward and group picture! (15 min)

Ticketing

We do not manage or distribute tickets on behalf of our satellite PCD events. As an organizer, you will determine whether ticketing is a preferable option for your event. We encourage all organizers who are planning to charge a fee for the event to provide affordable options to promote accessibility and inclusivity.

Create an event webpage

Create an event page on platforms such as Meetup or Eventbrite.

Update event detail on our website

Please visit the PCD @ Worldwide Map Spreadsheet to update your event details. This is important since your conference guests might visit our website to find out details about your event. Please contact day@processing.org if you did not receive a link to the spreadsheet.

Promotion

Promo Package

PCD @ Worldwide logos are created by Yuehao Jiang and can be downloaded online. The promo package is licensed under a Creative Commons Attribution 4.0 International license. Send us a copy if you end up remixing the graphics; we would love to see them!

Online Promotion Checklist

- Press release (sample press release on p. 280)
- Mailing lists
- Personal emails to friends and communities
- Social media - use hashtag #PCD2019 to connect with Processing Foundation and PCD communities around the world
- Blog posts - consider writing an in-depth post before / after PCD on platforms such as Medium to generate interest.
- Offline Promotion Checklist

- Flyers
- Posters
- Stickers

Share your PCD

We would love to receive pictures and documentations from your planning meetings and your Processing Community Day. Please share any materials you generate along the way with us via Twitter (@ProcessingOrg) or email (day@processing.org).

Sample Press Release

July 12, 2018, For Immediate Release

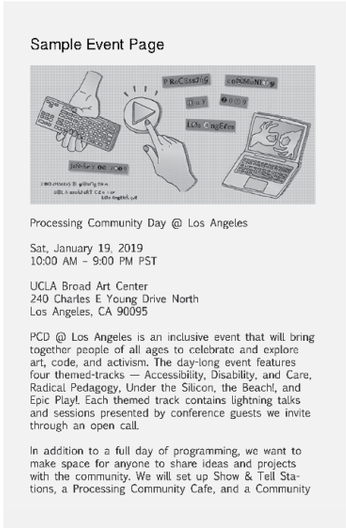
Join Processing Community Day to Celebrate Art, Code, and Diversity

Processing Foundation is pleased to announce Processing Community Day (PCD), a day to celebrate art, code, and diversity, held in Los Angeles and communities worldwide.

PCD @ Los Angeles is an inclusive event that will bring together people of all ages to celebrate and explore art, code, and activism. The day-long event features four themed tracks – Accessibility, Disability, and Care; Radical Pedagogy; Under the Silicon, the Beach!; and Epic Play! Each themed track contains lightning talks and sessions presented by conference guests we invite through an open call.

Processing is a free and open-source software platform for code within the context of the visual arts, created by Casey Reas and Ben Fry. It is complemented by a web version, called p5.js, created by Lauren McCarthy. To date, Processing and p5.js are used by a worldwide community of artists, coders, educators, and students.

Additional information on Processing Community Day can be found at day.processing.org., Xin Xin, Director & Lead Organizer, Processing Community Day 2019, Tel: +1 (323) 456-6613, day@processing.org, @processingOrg.


Sample Event Page

Processing Community Day @ Los Angeles

Sat, January 19, 2019
10:00 AM – 9:00 PM PST

UCLA Broad Art Center
240 Charles E Young Drive North
Los Angeles, CA 90095

PCD @ Los Angeles is an inclusive event that will bring together people of all ages to celebrate and explore art, code, and activism. The day-long event features four themed-tracks — Accessibility, Disability, and Care, Radical Pedagogy, Under the Silicon, the Beach!, and Epic Play!. Each themed track contains lightning talks and sessions presented by conference guests we invite through an open call.

In addition to a full day of programming, we want to make space for anyone to share ideas and projects with the community. We will set up Show & Tell Stations, a Processing Community Cafe, and a Community

The Processing Community Day Los Angeles web banner with sample text over a gray background.

Sample Event Page

Processing Community Day @ Los Angeles, Sat, January 19, 2019, 10:00 AM – 9:00 PM PST, UCLA Broad Art Center, 240 Charles E Young Drive North, Los Angeles, CA 90095

PCD @ Los Angeles is an inclusive event that will bring together people of all ages to celebrate and explore art, code, and activism. The day-long event features four themed-tracks – Accessibility, Disability, and Care, Radical Pedagogy, Under the Silicon, the Beach!, and Epic Play!. Each themed track contains lightning talks and sessions presented by conference guests we invite through an open call.

In addition to a full day of programming, we want to make space for anyone to share ideas and projects with the community. We will set up Show & Tell Stations, a Processing Community Cafe, and a Community Open Mic Session for participants to sign up on the day of the event. The program will wrap up with an afterparty consisting of performances, food, and drinks.

Tickets

Early-bird ticketing will open on September 1, 2018. Regular ticketing and PCD Scholarship application opens on November 1, 2018. Sign up for our newsletter to receive a reminder.

Every conference pass includes access to all lightning talks, entries to two workshops, performances, and the afterparty. The package also includes a lunch box (vegetarian / vegan options available), all-day coffee & snacks, and PCD / Processing / p5 combo stickers.

Children under 12 will gain free entries to PCD @ Los Angeles. Children's diet varies widely at different age, therefore we would like to ask you to either bring lunch boxes for your children on the day of the event, or order PCD @ Los Angeles' children's meals when purchasing your ticket.

Accessibility

All lightning talks will include ASL interpretation and be livestreamed over the internet. Our venues are wheelchair-accessible, and we will provide reserved seatings for attendees with mobility devices. Once ticketing opens, please fill out our ticketing survey so that we can learn more about your needs.



Parking & Transportation

The closest parking lot to UCLA Broad Art Center is Parking Structure 3. Visitor parking is $12 per day, and accessible parking for individuals with disabilities is $8 per day. We strongly recommend arranging carpool, using a ridesharing service or public transportation to commute to the venue.

PCD Scholarship

We are currently fundraising for the scholarship and will be announcing application details on November 1, 2018. PCD scholarship is made possible by PCD @ Los Angeles supporters and sponsors. Learn how you can support us.

Beginner Workshop

Examples

Play with p5, Provided by Lauren McCarthy

Setup (10 mins)

1. We will be using a new p5.js web editor. You can find it at alpha.editor.p5js.org.
2. Sign up for an account, and name and save your first project.
3. You are ready to begin creating with p5.js!

Experiment (45-90 mins)

One way to start learning p5.js is just to play with it and see what patterns you notice.

1. Try adding some of the lines of code from this list into the places where it says setup() and draw(). To do this, type the line as you see it, but replace each letter in pink with a number. Press the play button to run your code. Can you figure out what each of the lines does?
2. Try changing the numbers to see what happens. Press play each time to run your code again. Can you figure out what the different letters stand for?
3. Can you explain the difference between putting code in setup() and putting code in draw()?

```
createCanvas(w, h);
ellipse(x, y, w, h);
rect(x, y, w, h);
21 22
line(x1, y1, x2, y2);
triangle(x1, y1, x2, y2, x3, y3);
fill(g);
fill(r, g, b, a);
stroke(r, g, b, a);
strokeWeight(w);
background(r, g, b, a);
print(s);
// these ones below get put inside
of other functions above
// as replacements for the letters
mouseX
mouseY
random(max)
second()
```

4. After experimenting for a while individually, team up with a partner and see what you can figure out with your knowledge combined.
5. Finally, get together with the entire group and walk through what each line does, and the questions above. The instructor may lead this portion with their computer plugged into a projector.

Sketch (30+ minutes)

Taking what you learned in the previous activity, create a sketch of your own. One at a time, try to add the following:

1. One element that is different every time you run the program.
2. One element that changes with the mouse.
3. One element that changes over time.

The following resources may be helpful:

- p5js.org - p5.js website, includes library, reference, examples, and tutorials
- p5js.org/reference - p5.js reference

More Ideas

- Create a portrait of your neighbor. Use only 2D primitive shapes – arc(), curve(), ellipse(), line(), point(), quad(), rect(), triangle() – and basic color functions – background(), colorMode(), fill(), noFill(), noStroke(), stroke(). Remember to use createCanvas() to specify the dimensions of your canvas.
- Go for a walk and find an object that intrigues or inspires you. Replicate it with code. It can be as realistic or abstract as you like. Include one element that is different each time you load the sketch.
- Exquisite corpse. One person sets up a blank sketch with a canvas. They add one line of code, hit play, and pass their laptop to the person next to them. The next person adds one line of code, hits play, and passes the laptop on. Repeat until it has gone all the way around the room. This can be done simultaneously with more than one laptop, so multiple compositions are created in parallel. It may help to arrange desks in a circle for easier passing. When laptops have returned to their original owners, place all laptops open on the desks. Participants may walk slowly in a circle around them to view the exquisite sketches.

Going Further

The following resources are helpful for learning more about p5.js and coding.

- p5.js forum - good place for p5.js programming questions
- Daniel Shiffman's Coding Train Videos - p5.js video tutorials
- Getting Started with p5.js - by Lauren McCarthy, Casey Reas, and Ben Fry
- Intro to Programming for the Visual Arts with p5.js - online class, free with sign up
- Open Processing - a website for sharing p5.js sketches, getting inspiration, and learning from others

10PRINT Coding Challenge, Provided by Dan Shiffman

Inspired by the book 10 PRINT CHR$(205.5+RND(1)); : GOTO 10 by Nick Montfort, Patsy Baudoin, John Bell, Ian Bogost, Jeremy Douglass, Mark C. Marino, Michael Mateas, Casey Reas, Mark Sample, and Noah Vawter. This book is about a one-line Commodore 64 BASIC program, published in November 2012. Book purchases support the nonprofit organizations The Electronic Literature Organization (to which all royalties are being donated) and The MIT Press, the book's publisher. http://10print.org

Setup (10 mins)

This tutorial uses the new p5.js web editor. You can find it at alpha.editor.p5js.org. Sign up for an account, and name and save your first project. Alternatively, you can use the Processing IDE. Download at https://processing.org/download/. Launch and you are ready to begin! You are ready to begin creating your own 10 PRINT design!

Video (10 mins)

The Coding Train YouTube channel includes many coding challenge videos that walk through the process of creating a p5 (or Processing) sketch. In this activity we'll build off of the 10PRINT Coding Challenge: https://youtu.be/bEyTZ5ZZxZs.

You can also watch this video to see the output of the original Commodore 64 program: https://youtu.be/m9joBLOZVEo.

Sketch (30+ minutes)

Open the code from the video. p5.js code: https://alpha.editor.p5js.org/codingtrain/sketches/ryxWYgmwX. Processing code: https://goo.gl/k6q1L7. Ideas for variations: What other shapes or colors can you draw instead of a line? What happens if you vary the probability? What do you do when the design reaches the bottom of the page? Can you get it to scroll continuously?

Processing release 0001 was built August 9, 2001 and release 1277 (4.0 beta 2) was built October 4, 2021, but the original Processing software is only the beginning of what has developed over the years. There are currently over 40 active software repositories on the Processing Github account with contributions from hundreds of people over the many years of development. The pages that follow document the release dates and version numbers for a selection of the Processing Foundation software releases.

REL Processing, Recorded 2001–2021.

001-1 Date: Aug 9, 2001 Version: 0001

001-2 Date: Aug 10, 2001 Version: 0002

001-3 Date: Aug 10, 2001 Version: 0003

001-4 Date: Sep 12, 2001 Version: 0004

001-5 Date: Sep 17, 2001 Version: 0005

001-6 Date: Sep 17, 2001 Version: 0006

001-7 Date: Sep 19, 2001 Version: 0007

001-8 Date: Sep 26, 2001 Version: 0008

001-9 Date: Sep 28 2001 Version: 0009

001-10 Date: Nov 28, 2001 Version: 0010

001-11 Date: Nov 28, 2001 Version: 0011

001-12 Date: Nov 29, 2001 Version: 0012

001-13 Date: Unlisted Version: 0013

001-14 Date: Dec 10, 2001 Version: 0014

001-15 Date: Dec 11, 2001 Version: 0015

001-16 Date: Jan 20, 2002 Version: 0016

001-17 Date: Jan 23, 2002 Version: 0017

001-18 Date: Feb 11, 2002 Version: 0018

001-19 Date: Feb 25, 2002 Version: 0019

001-20 Date: Unlisted Version: 0020

001-21 Date: Apr 22, 2002 Version: 0023

001-22 Date: Apr 22, 2002 Version: 0024

001-23 Date: Jun 6, 2002 Version: 0025

001-24 Date: Jun 7, 2002 Version: 0026

001-25 Date: Jun 7, 2002 Version: 0027

001-26 Date: Jul 1, 2002 Version: 0028

001-27 Date: Jul 1, 2002 Version: 0029

001-28 Date: Jul 2, 2002 Version: 0030

001-29 Date: Jul 5, 2002 Version: 0031

001-30 Date: Jul 7, 2002 Version: 0032

001-31 Date: Jul 8, 2002 Version: 0033

001-32 Date: Jul 22, 2002 Version: 0034

001-33 Date: Jul 23, 2002 Version: 0035

001-12 Date: Jul 27, 2002 Version: 0036

001-13 Date: Jul 27, 2002 Version: 0037

001-14 Date: Jul 30, 2002 Version: 0038

001-15 Date: Jul 31, 2002 Version: 0039

001-16 Date: Aug 1, 2002 Version: 0040

001-17 Date: Aug 1, 2002 Version: 0041

001-18 Date: Aug 2, 2002 Version: 0042

001-19 Date: Aug 2, 2002 Version: 0043

001-30 Date: Sep 2, 2002 Version: 0044

001-31 Date: Oct 20, 2002 Version: 0045

001-32 Date: Nov 20, 2002 Version: 0046

001-33 Date: Jan 5, 2003 Version: 0047

001-34 Date: Jan 7, 2003 Version: 0048

001-35 Date: Jan 26, 2003 Version: 0049

001-36 Date: Jan 27, 2003 Version: 0050

001-37 Date: Jan 31, 2003 Version: 0051

001-38 Date: Feb 2, 2004 Version: 0068

001-39 Date: Mar 28, 2004 Version: 0069

001-40 Date: Sep 29, 2004 Version: 0070

001-41 Date: Unlisted Version: 0071

001-42 Date: Unlisted Version: 0072

001-43 Date: Unlisted Version: 0073

001-44 Date: Unlisted Version: 0074

001-45 Date: Unlisted Version: 0075

001-46 Date: Unlisted Version: 0076

001-47 Date: Mar 1, 2005 Version: 0077

001-48 Date: Mar 30, 2005 Version: 0078

001-49 Date: Mar 31, 2005 Version: 0079

001-50 Date: Apr 2, 2005 Version: 0080

001-51 Date: Apr 4, 2005 Version: 0081

001-52 Date: Apr 6, 2005 Version: 0082

001-53 Date: Apr 17, 2005 Version: 0083

001-54 Date: Apr 19, 2005 Version: 0084

001-55 Date: Apr 19, 2005 Version: 0085

001-56 Date: May 1, 2005 Version: 0086



001-57 Date: May 2, 2005 Version: 0087

001-58 Date: May 7, 2005 Version: 0088

001-59 Date: May 8, 2005 Version: 0089

001-60 Date: May 9, 2005 Version: 0090

001-61 Date: Jun 6, 2005 Version: 0091

001-62 Date: Jun 30, 2005 Version: 0092

001-63 Date: Oct 14, 2005 Version: 0093

001-64 Date: Oct 31, 2005 Version: 0094

001-65 Date: Nov 2, 2005 Version: 0095

001-66 Date: Nov 26, 2005 Version: 0096

001-67 Date: Nov 29, 2005 Version: 0097

001-68 Date: Dec 1, 2005 Version: 0098

001-69 Date: Dec 18, 2005 Version: 0099

001-70 Date: Jan 13, 2006 Version: 0100

001-71 Date: Jan 14, 2006 Version: 0101

001-72 Date: Jan 15, 2006 Version: 0102

001-73 Date: Jan 18, 2006 Version: 0103

001-74 Date: Feb 27, 2006 Version: 0104

001-75 Date: Feb 27, 2006 Version: 0105

001-76 Date: Mar 1, 2006 Version: 0106

001-77 Date: Mar 2, 2006 Version: 0107

001-78 Date: Mar 15, 2006 Version: 0108

001-79 Date: Mar 21, 2006 Version: 0109

001-80 Date: Mar 22, 2006 Version: 0110

001-81 Date: Mar 28, 2006 Version: 0111

001-82 Date: Apr 13, 2006 Version: 0112

001-83 Date: Apr 14, 2006 Version: 0113

001-84 Date: Apr 19, 2006 Version: 0114

001-85 Date: May 11, 2006 Version: 0115

001-86 Date: Sep 26, 2006 Version: 0116

001-87 Date: Oct 1, 2006 Version: 0117

001-88 Date: Oct 1, 2006 Version: 0118

001-89 Date: Oct 10, 2006 Version: 0119

001-90 Date: Nov 5, 2006 Version: 0120

001-91 Date: Nov 6, 2006 Version: 0121

001-92 Date: Nov 28, 2006 Version: 0122

001-93 Date: Nov 30, 2006 Version: 0123

001-94 Date: Feb 4, 2007 Version: 0124

001-95 Date: Jul 15, 2007 Version: 0125

001-96 Date: Oct 9, 2007 Version: 0126

001-97 Date: Oct 14, 2007 Version: 0127

001-98 Date: Oct 16, 2007 Version: 0128

001-99 Date: Oct 18, 2007 Version: 0129

001-100 Date: Oct 20, 2007 Version: 0130

001-101 Date: Oct 22, 2007 Version: 0131

001-102 Date: Oct 22, 2007 Version: 0132

001-103 Date: Oct 26, 2007 Version: 0133

001-104 Date: Nov 17, 2007 Version: 0134

001-105 Date: Nov 21, 2007 Version: 0135

001-106 Date: May 29, 2008 Version: 0136

001-107 Date: May 30, 2008 Version: 0137

001-108 Date: May 31, 2008 Version: 0138

001-109 Date: Jun 4, 2008 Version: 0139

001-110 Date: Jun 10, 2008 Version: 0140

001-111 Date: Jun 11, 2008 Version: 0141

001-112 Date: Jun 16, 2008 Version: 0142

001-113 Date: Jun 28, 2008 Version: 0143

001-114 Date: Aug 9, 2008 Version: 0144

001-115 Date: Aug 15, 2008 Version: 0145

001-116 Date: Aug 16, 2008 Version: 0146

001-117 Date: Aug 18, 2008 Version: 0147

001-118 Date: Aug 19, 2008 Version: 0148

001-119 Date: Oct 15, 2008 Version: 0149

001-120 Date: Oct 15, 2008 Version: 0150

001-121 Date: Oct 17, 2008 Version: 0151

001-122 Date: Oct 19, 2008 Version: 0152

001-123 Date: Oct 21, 2008 Version: 0153

001-124 Date: Oct 21, 2008 Version: 0154

001-125 Date: Nov 4, 2008 Version: 0155

001-126 Date: Nov 10, 2008
Version: 0156
001-127 Date: Nov 18, 2008
Version: 0157
001-128 Date: Nov 22, 2008
Version: 0158
001-129 Date: Nov 24, 2008
Version: 1.0
001-130 Date: Nov 29, 2008
Version: 1.0.1
001-131 Date: Feb 21, 2009
Version: 1.0.2
001-132 Date: Feb 24, 2009
Version: 1.0.3
001-133 Date: May 31, 2009
Version: 1.0.3
001-134 Date: Jun 7, 2009
Version: 1.0.5
001-135 Date: Aug 12, 2009
Version: 1.0.6
001-136 Date: Sep 4, 2009
Version: 1.0.7
001-137 Date: Oct 18, 2009
Version: 1.0.8
001-138 Date: Oct 20, 2009
Version: 1.0.9
001-139 Date: Mar 11, 2010
Version: 1.1
001-140 Date: Jul 13, 2010
Version: 1.2
001-141 Date: Jul 14, 2010
Version: 1.2.1
001-142 Date: Apr 17, 2011
Version: 1.5
001-143 Date: May 15, 2011
Version: 1.5.1
001-144 Date: Oct 31, 2011
Version: 2.0a2
001-145 Date: Nov 5, 2011
Version: 2.0a3
001-146 Date: Nov 10, 2011
Version: 2.0a4
001-147 Date: Mar 23, 2012
Version: 2.0a5
001-148 Date: Jun 1, 2012
Version: 2.0a6
001-149 Date: Jul 29, 2012
Version: 2.0a7
001-150 Date: Aug 5, 2012
Version: 2.0a8
001-151 Date: Sep 1, 2012
Version: 2.0a9
001-152 Date: Sep 3, 2012
Version: 2.0b1
001-153 Date: Sep 7, 2012
Version: 2.0b2
001-154 Date: Sep 10, 2012
Version: 2.0b3
001-155 Date: Oct 21, 2012
Version: 2.0b4
001-156 Date: Oct 22, 2012
Version: 2.0b5
001-157 Date: Nov 2, 2012
Version: 2.0b6
001-158 Date: Dec 7, 2012
Version: 2.0b7
001-159 Date: Feb 24, 2013
Version: 2.0b8
001-160 Date: May 18, 2013
Version: 2.0b9
001-161 Date: Jun 3, 2013
Version: 2.0
001-162 Date: Jun 21, 2013
Version: 2.0.1
001-163 Date: Jun 21, 2013
Version: 2.0.2
001-164 Date: Sep 5, 2013
Version: 2.0.3
001-165 Date: Oct 20, 2013
Version: 2.1 beta 1
001-166 Date: Oct 27, 2013
Version: 2.1
001-167 Date: Jan 21, 2014
Version: 2.1.1
001-168 Date: Apr 15, 2014
Version: 2.1.2
001-169 Date: May 12, 2014
Version: 2.2
001-170 Date: May 19, 2014
Version: 2.2.1
001-171 Date: Jul 16, 2014
Version: 3.0a1
001-172 Date: Jul 31, 2014
Version: 3.0a2
001-173 Date: Aug 26, 2014
Version: 3.0a3
001-174 Date: Sep 12, 2014
Version: 3.0a4
001-175 Date: Nov 16, 2014
Version: 3.0a5
001-176 Date: Apr 25, 2015
Version: 3.0a6
001-177 Date: Apr 26, 2015
Version: 3.0a7
001-178 Date: May 17, 2015
Version: 3.0a8
001-179 Date: May 19, 2015
Version: 3.0a9
001-180 Date: Jun 9, 2015
Version: 3.0a10
001-181 Date: Jul 16, 2015
Version: 3.0a11
001-182 Date: Aug 6, 2015
Version: 3.0b1
001-183 Date: Aug 9, 2015
Version: 3.0b2
001-184 Date: Aug 11, 2015
Version: 3.0b3
001-185 Date: Aug 17, 2015
Version: 3.0b4
001-186 Date: Aug 24, 2015
Version: 3.0b5
001-187 Date: Sep 11, 2015
Version: 3.0b6
001-188 Date: Sep 22, 2015
Version: 3.0b7
001-189 Date: Sep 30, 2015
Version: 3.0
001-190 Date: Oct 23, 2015
Version: 3.0.1
001-191 Date: May 8, 2016
Version: 3.1
001-192 Date: May 6, 2016
Version: 3.1.1
001-193 Date: Jul 29, 2016
Version: 3.1.2
001-194 Date: Aug 16, 2016
Version: 3.2



001-195 Date: Oct 30, 2016
Version: 3.2.2
001-196 Date: Nov 7, 2016
Version: 3.2.3
001-197 Date: Jan 29, 2017
Version: 3.2.4
001-198 Date: Feb 12, 2017
Version: 3.3
001-199 Date: Apr 23, 2017
Version: 3.3.1
001-200 Date: Apr 25, 2017
Version: 3.3.2
001-201 Date: May 2, 2017
Version: 3.3.3
001-202 Date: Jun 3, 2017
Version: 3.3.4
001-203 Date: Jun 23, 2017
Version: 3.3.5
001-204 Date: Sep 4, 2017
Version: 3.3.6
001-205 Date: Mar 13, 2018
Version: 3.3.7
001-206 Date: Jul 26, 2018
Version: 3.4
001-207 Date: Jan 20, 2019
Version: 3.5
001-208 Date: Jan 21, 2019
Version: 3.5.1
001-209 Date: Jan 22, 2019
Version: 3.5.2
001-210 Date: Feb 3, 2019
Version: 3.5.3
001-211 Date: Jan 17, 2020
Version: 3.5.4
001-212 Date: 18 January 2019
Version: 4.0 alpha 1
001-213 Date: 15 September 2020
Version: 4.0 alpha 2
001-214 Date: 17 January 2021
Version: 4.0 alpha 3
001-215 Date: 15 June 2021
Version: 4.0 alpha 4
001-216 Date: 24 June 2021
Version: 4.0 alpha 5
001-217 Date: 10 July 2021
Version: 4.0 alpha 6
001-218 Date: 9 August 2021
Version: 4.0 beta 1
001-219 Date: 4 October 2021
Version: 4.0 beta 2

REL p5.js, Recorded 2014–2021.
002-1 Date: Jul 21, 2014
Version: 0.2.22
002-2 Date: Jul 31, 2014
Version: 0.2.23
002-3 Date: Aug 6, 2014
Version: 0.3.0 (first public beta release)
002-4 Date: Aug 8, 2014
Version: 0.3.2
002-5 Date: Aug 17, 2014
Version: 0.3.3
002-6 Date: Aug 26, 2014
Version: 0.3.4
002-7 Date: Aug 28, 2014
Version: 0.3.5
002-8 Date: Sep 11, 2014
Version: 0.3.6
002-9 Date: Sep 19, 2014
Version: 0.3.7
002-10 Date: Sep 25, 2014
Version: 0.3.8
002-11 Date: Oct 10, 2014
Version: 0.3.9
002-12 Date: Nov 3, 2014
Version: 0.3.11
002-13 Date: Nov 23, 2014
Version: 0.3.12
002-14 Date: Dec 4, 2014
Version: 0.3.13
002-15 Date: Dec 10, 2014
Version: 0.3.14
002-16 Date: Dec 23, 2014
Version: 0.3.15
002-17 Date: Jan 8, 2015
Version: 0.3.16
002-18 Date: Feb 2, 2015
Version: 0.4.0
002-19 Date: Feb 13, 2015
Version: 0.4.1
002-20 Date: Feb 16, 2015
Version: 0.4.2
002-21 Date: Mar 30, 2015
Version: 0.4.3
002-22 Date: Apr 12, 2015
Version: 0.4.4
002-23 Date: May 27, 2015
Version: 0.4.5
002-24 Date: Jun 24, 2015
Version: 0.4.6
002-25 Date: Jul 17, 2015
Version: 0.4.7
002-26 Date: Aug 18, 2015
Version: 0.4.8
002-27 Date: Aug 31, 2015
Version: 0.4.9
002-28 Date: Sep 12, 2015
Version: 0.4.10
002-29 Date: Sep 14, 2015
Version: 0.4.11
002-30 Date: Sep 15, 2015
Version: 0.4.12
002-31 Date: Sep 20, 2015
Version: 0.4.13
002-32 Date: Oct 5, 2015
Version: 0.4.14
002-33 Date: Oct 6, 2015
Version: 0.4.15
002-34 Date: Oct 12, 2015
Version: 0.4.16
002-35 Date: Oct 13, 2015
Version: 0.4.17
002-36 Date: Nov 9, 2015
Version: 0.4.18
002-37 Date: Nov 11, 2015
Version: 0.4.19
002-38 Date: Dec 11, 2015
Version: 0.4.20
002-39 Date: Jan 4, 2016
Version: 0.4.21
002-40 Date: Feb 4, 2016
Version: 0.4.22
002-41 Date: Mar 4, 2016
Version: 0.4.23
002-42 Date: Apr 25, 2016
Version: 0.4.24

002-43 Date: May 2, 2016
Version: 0.5.0

002-44 Date: Jun 10, 2016
Version: 0.5.1

002-45 Date: Jun 16, 2016
Version: 0.5.2

002-46 Date: Aug 17, 2016
Version: 0.5.3

002-47 Date: Oct 1, 2016
Version: 0.5.4

002-48 Date: Dec 5, 2016
Version: 0.5.5

002-49 Date: Jan 3, 2017
Version: 0.5.6

002-50 Date: Feb 8, 2017
Version: 0.5.7

002-51 Date: Mar 25, 2017
Version: 0.5.8

002-52 Date: May 10, 2017
Version: 0.5.9

002-53 Date: May 18, 2017
Version: 0.5.10

002-54 Date: Jun 1, 2017
Version: 0.5.11

002-55 Date: Aug 11, 2017
Version: 0.5.12

002-56 Date: Aug 25, 2017
Version: 0.5.13

002-57 Date: Aug 28, 2017
Version: 0.5.14

002-58 Date: Oct 9, 2017
Version: 0.5.15

002-59 Date: Oct 11, 2017
Version: 0.5.16

002-60 Date: Jan 19, 2018
Version: 0.6.0

002-61 Date: Apr 27, 2018
Version: 0.6.1

002-62 Date: Aug 9, 2018
Version: 0.7.0

002-63 Date: Aug 10, 2018
Version: 0.7.1

002-64 Date: Sep 2, 2018
Version: 0.7.2

002-65 Date: Jan 20, 2019
Version: 0.7.3

002-66 Date: Apr 8, 2019
Version: 0.8.0

002-67 Date: Jul 1, 2019
Version: 0.9.0

002-68 Date: Oct 14, 2019
Version: 0.10.0

002-69 Date: Oct 14, 2019
Version: 0.10.2

002-70 Date: Feb 29, 2020
Version: 1.0.0

002-71 Date: Jul 22, 2020
Version: 1.1.3

002-72 Date: Jul 22, 2020
Version: 1.1.4

002-73 Date: Jul 22, 2020
Version: 1.1.5

002-74 Date: Jul 22, 2020
Version: 1.1.7

002-75 Date: Jul 22, 2020
Version: 1.1.9

002-76 Date: Dec 20, 2020
Version: 1.2.0

002-77 Date: Mar 9, 2021
Version: 1.3.0

002-78 Date: Mar 28, 2021
Version: 1.3.1

002-79 Date: Jun 29, 2021
Version: 1.4.0

REL p5.js Editor (v1), Recorded 2014–2016.

003-1 Date: Jul 29, 2014
Version: 0.1.0
(p5.js IDE pre-release)

003-2 Date: Jul 30, 2014
Version: 0.1.1

003-3 Date: Jul 31, 2014
Version: 0.1.3

003-4 Date: Jul 31, 2014
Version: 0.1.4

003-5 Date: Aug 3, 2014
Version: 0.1.6

003-6 Date: Aug 11, 2014
Version: 0.1.7

003-7 Date: Sep 18, 2014
Version: 0.1.8

003-8 Date: May 30, 2015
Version: 0.3.0

003-9 Date: Jul 11, 2015
Version: 0.3.1

003-10 Date: Jul 13, 2015
Version: 0.3.2

003-11 Date: Aug 5, 2015
Version: 0.4.0

003-12 Date: Aug 31, 2015
Version: 0.5.0

003-13 Date: Sep 2, 2015
Version: 0.5.1

003-14 Date: Nov 2, 2015
Version: 0.5.6

003-15 Date: Dec 11, 2015
Version: 0.5.7

003-16 Date: Mar 4, 2016
Version: 0.5.8

003-17 Date: Mar 4, 2016
Version: 0.5.9

003-18 Date: Mar 14, 2016
Version: 0.5.10

003-19 Date: Jun 21, 2016
Version: 0.6.0

003-20 Date: Aug 18, 2016
Version: 0.6.1

003-21 Date: Nov 10, 2016
Version: 0.6.2

REL p5.js Editor (v2), Recorded 2020–2021.

004-1 Date: May 26, 2020
Version: v1.0.0

004-2 Date: May 26, 2020
Version: v1.0.1

004-3 Date: Jun 1, 2020
Version: v1.0.2

004-4 Date: Jun 10, 2020
Version: v1.0.3

004-5 Date: Jul 13, 2020
Version: v1.0.4

004-6 Date: Jul 13, 2020
Version: v1.0.5

004-7 Date: Jul 13, 2020
Version: 1.0.5



004-8 Date: Jul 20, 2020
Version: v1.0.6

004-9 Date: Aug 6, 2020
Version: v1.0.7

004-10 Date: Sep 11, 2020
Version: v1.0.8

004-11 Date: Sep 29, 2020
Version: v1.1.0

004-12 Date: Sep 29, 2020
Version: v1.1.1

004-13 Date: Nov 17, 2020
Version: v1.2.0

004-14 Date: Nov 17, 2020
Version: v1.2.1

004-15 Date: Nov 17, 2020
Version: v1.2.2

004-16 Date: Mar 5, 2021
Version: v1.3.0

004-17 Date: May 7, 2021
Version: v1.4.0

004-18 Date: Jun 14, 2021
Version: v1.4.1

004-19 Date: Jun 28, 2021
Version: v1.4.2

004-20 Date: Jul 11, 2021
Version: v2.0.0

004-21 Date: Jul 12, 2021
Version: v2.0.1

004-22 Date: Jul 15, 2021
Version: v2.0.2

004-23 Date: Sep 8, 2021
Version: v2.0.3

004-24 Date: Oct 5, 2021
Version: v2.1.0

004-25 Date: Oct 8, 2021
Version: v2.1.1

REL p5.Sound, 2014–2021.

005-1 Date: Aug 22, 2014
Version: 0.14

005-2 Date: Feb 4, 2016
Version: 0.14

005-3 Date: Feb 28, 2017
Version: 0.3.5

005-4 Date: Jul 15, 2018
Version: 0.3.8

005-5 Date: Sep 8, 2018
Version: 0.3.9

005-6 Date: Jan 10, 2019
Version: 0.3.10

005-7 Date: Apr 1, 2020
Version: 0.3.12

005-8 Date: May 25, 2021
Version: 1.0.0

005-9 Date: May 25, 2021
Version: 1.0.1

REL Processing for Android, Recorded 2014–2021.

006-1 Date: Sep 11, 2014
Version: Android Mode 0228

006-2 Date: Nov 5, 2014
Version: Android Mode 0232

006-3 Date: Jul 12, 2015
Version: Android Mode 0238

006-4 Date: Jul 17, 2015
Version: Android Mode 0239

006-5 Date: Aug 20, 2015
Version: Android Mode 0241

006-6 Date: Aug 25, 2015
Version: Android Mode 0242

006-7 Date: Sep 12, 2015
Version: Android Mode 0243

006-8 Date: Sep 26, 2015
Version: Android Mode 0244

006-9 Date: Oct 4, 2015
Version: Android Mode 0245

006-10 Date: Nov 2, 2015
Version: Android Mode 0246

006-11 Date: Mar 5, 2016
Version: Android Mode 0247

006-12 Date: Apr 5, 2016
Version: Android Mode 0248

006-13 Date: Apr 16, 2016
Version: Android Mode 0249

006-14 Date: May 9, 2016
Version: Android Mode 0250

006-15 Date: Jun 10, 2016
Version: Android Mode 0251

006-16 Date: Jun 10, 2016
Version: Android Mode 0254

006-17 Date: Jul 20, 2016
Version: Android Mode 0252

006-18 Date: Jul 22, 2016
Version: Android Mode 0255

006-19 Date: Dec 30, 2016
Version: Android Mode 0253

006-20 Date: Jan 4, 2017
Version: Android Mode 0256

006-21 Date: Feb 10, 2017
Version: Android Mode 0257

006-22 Date: Mar 9, 2017
Version: Android Mode 0258

006-23 Date: Apr 4, 2017
Version: Android Mode 0259

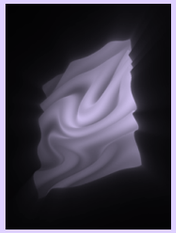
006-24 Date: Apr 11, 2017
Version: Android Mode 0260

006-25 Date: May 6, 2017
Version: Android Mode 0261

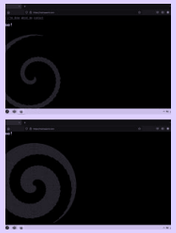
006-26 Date: May 26, 2017
Version: Android
Mode 0262
006-27 Date: Jul 15, 2017
Version: Android
Mode 0263
006-28 Date: Aug 6, 2017
Version: Android
Mode 0264
006-29 Date: Aug 12, 2017
Version: Android
Mode 0265
006-30 Date: Aug 21, 2017
Version: Android
Mode 0266
006-31 Date: Sep 1, 2017
Version: Android
Mode 0267
006-32 Date: Jan 28, 2018
Version: Android
Mode 0268
006-33 Date: Jun 19, 2018
Version: Android
Mode 0269
006-34 Date: Aug 16, 2018
Version: Android
Mode 0270
006-35 Date: Dec 28, 2018
Version: Android
Mode 0271
006-36 Date: Jan 5, 2019
Version: Android
Mode 0272
006-37 Date: Feb 15, 2019
Version: Android
Mode 0273
006-38 Date: Jun 2, 2019
Version: Android
Mode 0274
006-39 Date: Jul 1, 2019
Version: Android
Mode 0275
006-40 Date: Oct 29, 2019
Version: Android
Mode 0276
006-41 Date: Nov 4, 2019
Version: Android
Mode 0277
006-42 Date: Jan 3, 2021
Version: Android
Mode 0278
006-43 Date: Apr 29, 2021
Version: Android
Mode 0279
006-44 Date: Apr 29, 2021
Version: 4.2.1
006-45 Date: Aug 12, 2021
Version: 4.5.0 Alpha 1
006-46 Date: Aug 18, 2021
Version: 4.5.0 Alpha 2

CON 001

CON 002

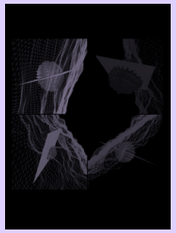


CON 003



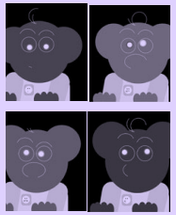
CON 004

CON 005



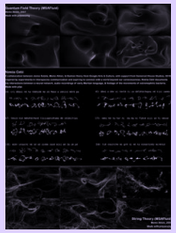
CON 006


CON 007



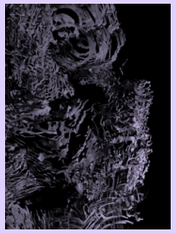
CON 008


CON 009

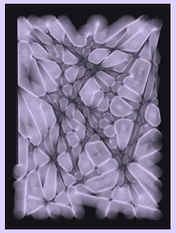


CON 010


CON 011



CON 012

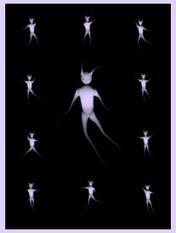


CON 013

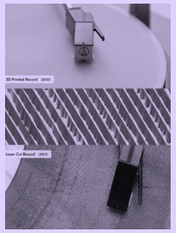

CON 014

CON 015

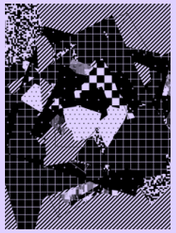

CON 016



CON 017

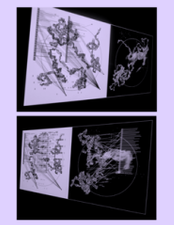


CON 018



CON 019

CON 020

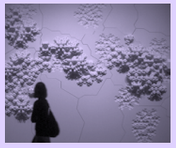


CON 021

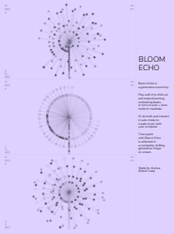
CON 022


CON 023

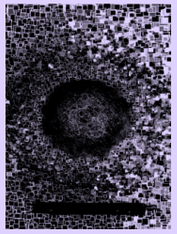
CON 024




CON 025

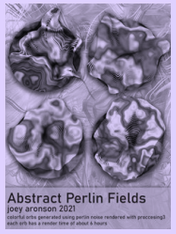



CON 026



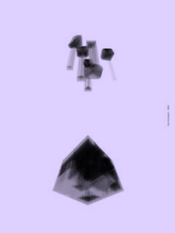
CON 027

CON 028




CON 029

CON 030





CON 031

CON 032

CON 033

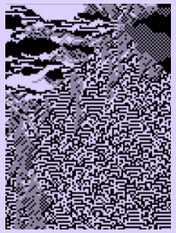

CON 034


CON 035

CON 036

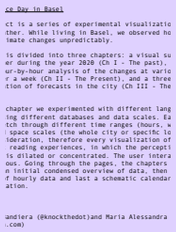

CON 037


CON 038



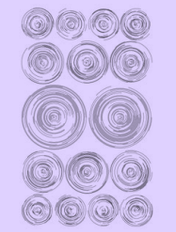
CON 039


CON 040

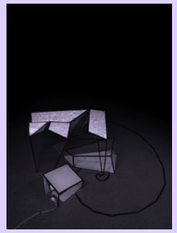


CON 041

CON 042



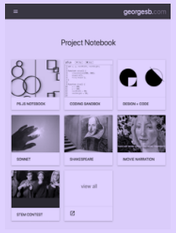
CON 043



CON 044

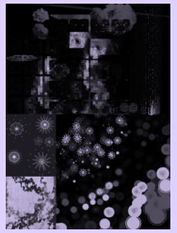
CON 045

CON 046

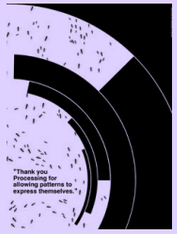



CON 047

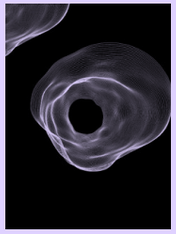
CON 048



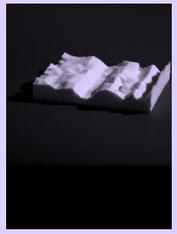
CON 049



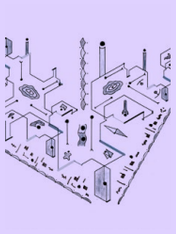

CON 050



CON 051

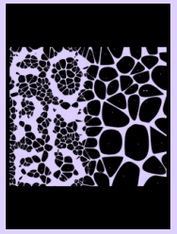


CON 052



CON 053


CON 054



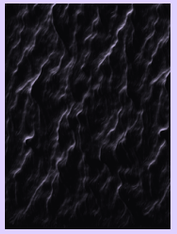
CON 055


CON 056

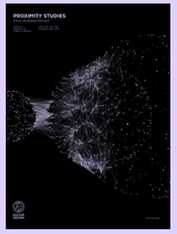

CON 057

CON 058

CON 059

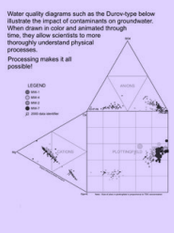


CON 060



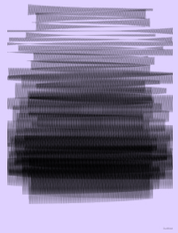
CON 061

CON 062




CON 063

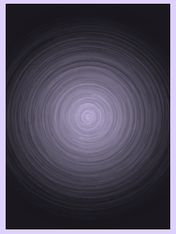

CON 064



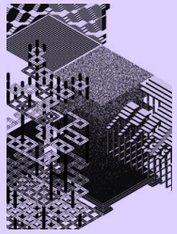
CON 065

CON 066

CON 067

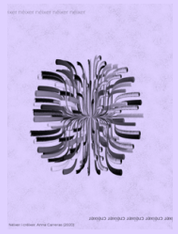


CON 068

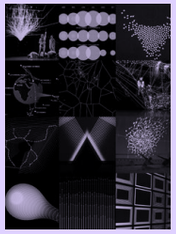


CON 069

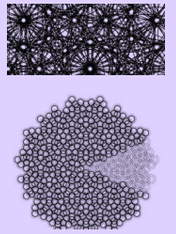
CON 070



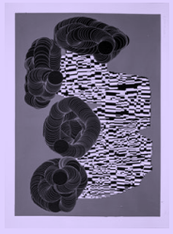
CON 071



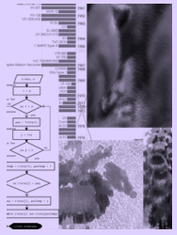
CON 072



CON 073



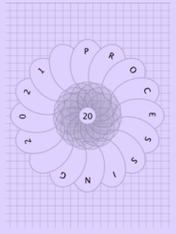
CON 074



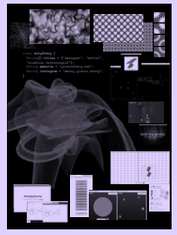
CON 075


CON 076


CON 077

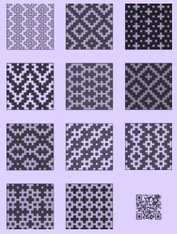



CON 078



CON 079


CON 080



CON 081

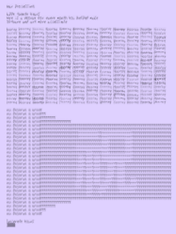
CON 082

CON 083


CON 084

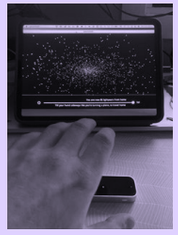

CON 085

CON 086



CON 087


CON 088

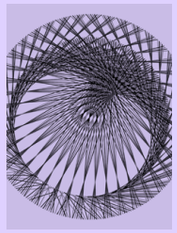


CON 089

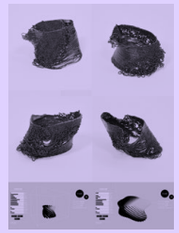
CON 090



CON 091



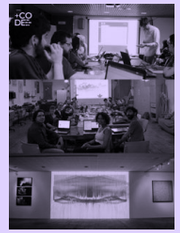
CON 092



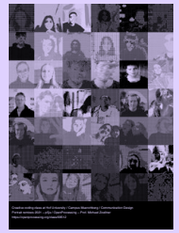
CON 093

CON 094

CON 095

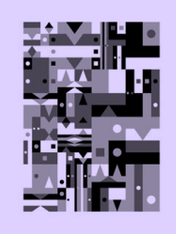


CON 096

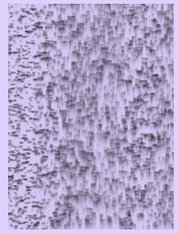


CON 097


CON 098



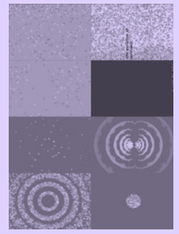
CON 099



CON 100

CON 101

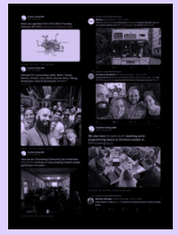
CON 102



CON 103

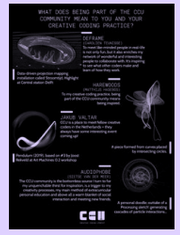

CON 104

CON 105



CON 106


CON 107



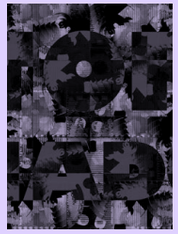

CON 108


CON 109


CON 110

CON 111

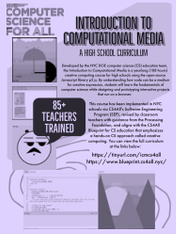

CON 112



CON 113

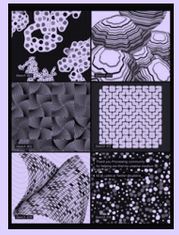

CON 114


CON 115



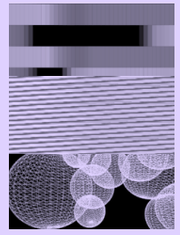

CON 116


CON 117



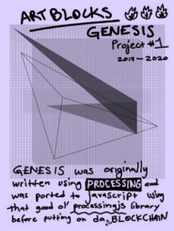
CON 118


CON 119



CON 120

CON 121




CON 122

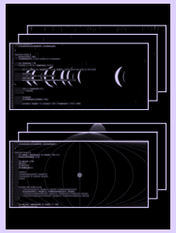

CON 123

CON 124

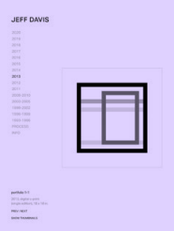
CON 125


CON 126

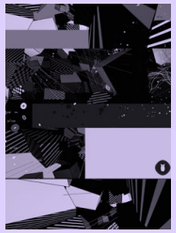
CON 127



CON 128

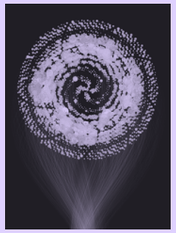



CON 129

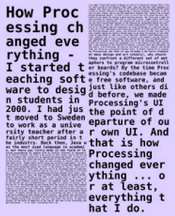


CON 130

CON 131

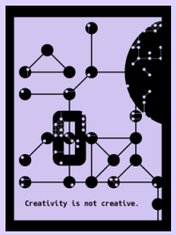


CON 132




CON 133

CON 134



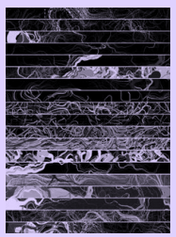

CON 135

CON 136

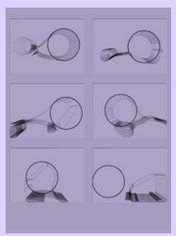
CON 137

CON 138

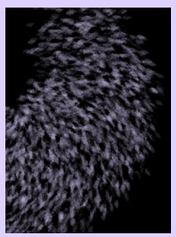

CON 139



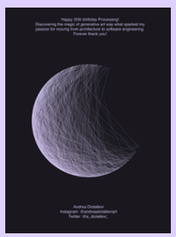
CON 140



CON 141



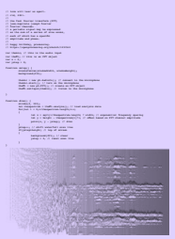
CON 142




CON 143

CON 144

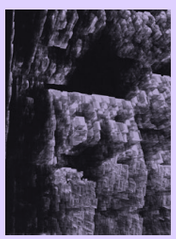
CON 145



CON 146

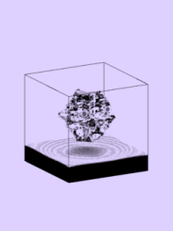
CON 147

CON 148



CON 149

CON 150





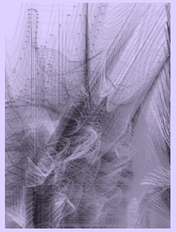
CON 151

CON 152


CON 153


CON 154

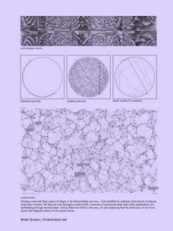

CON 155



CON 156


CON 157


CON 158



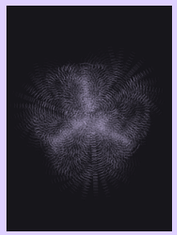
CON 159


CON 160


CON 161

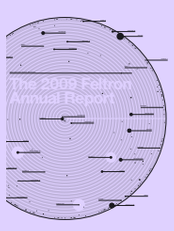

CON 162


CON 163



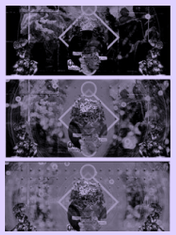
CON 164

CON 165

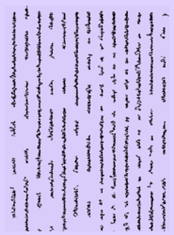



CON 166

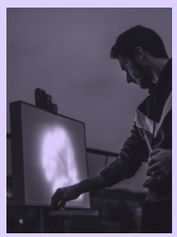
CON 167



CON 168



CON 169

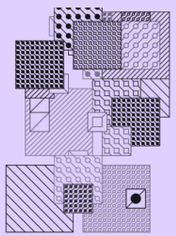


CON 170

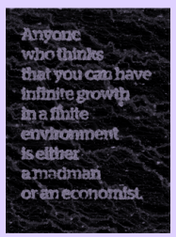
CON 171

CON 172


CON 173

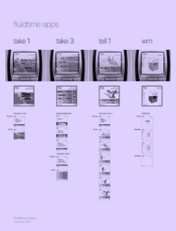


CON 174

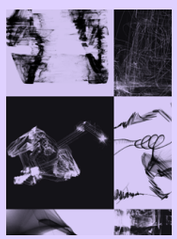



CON 175

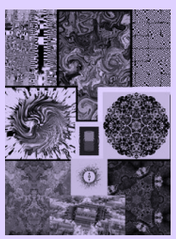
CON 176




CON 177



CON 178

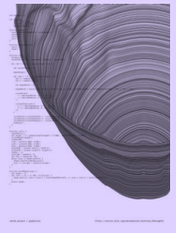


CON 179

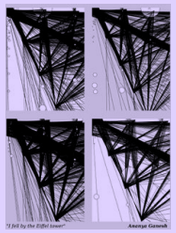
CON 180

CON 181

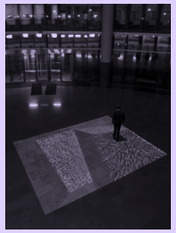
CON 182



CON 183

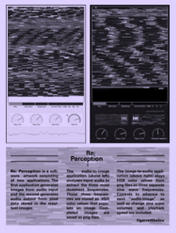


CON 184



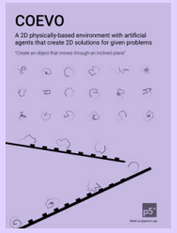
CON 185


CON 186

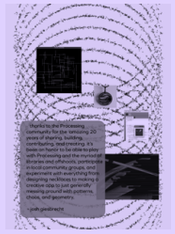


CON 187

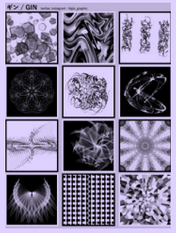
CON 188



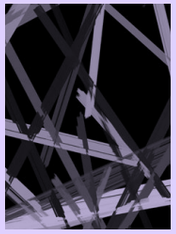

CON 189



CON 190



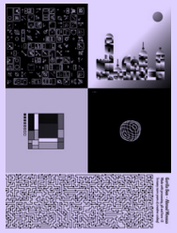
CON 191



CON 192

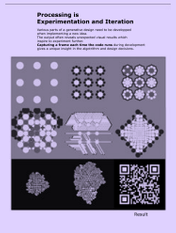

CON 193

CON 194

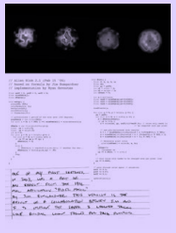


CON 195


CON 196

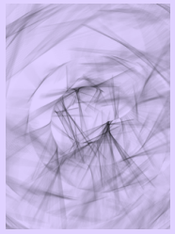



CON 197

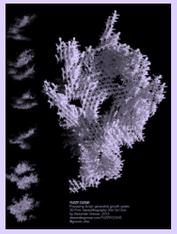


CON 198


CON 199



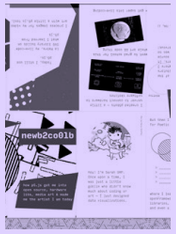
CON 200



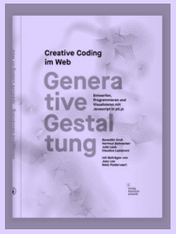
CON 201


CON 202

CON 203

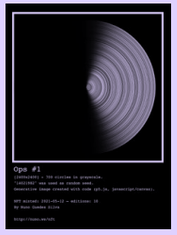



CON 204

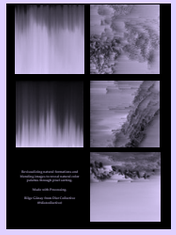



CON 205

CON 206

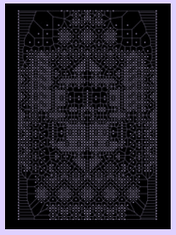



CON 207

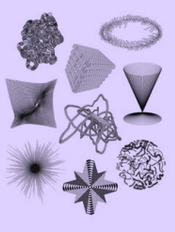


CON 208

CON 209

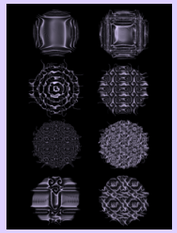


CON 210

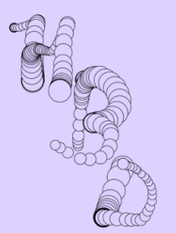




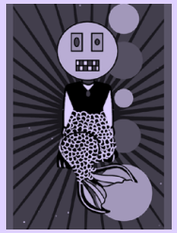
CON 211



CON 212



CON 213

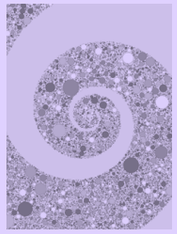


CON 214


CON 215

CON 216

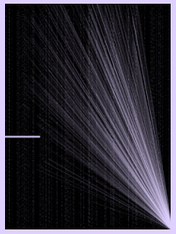
CON 217



CON 218

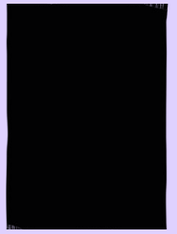

CON 219

CON 220



CON 221

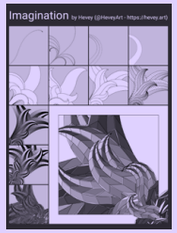
CON 222



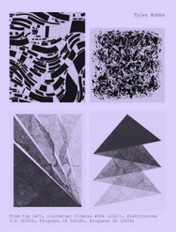
CON 223

CON 224

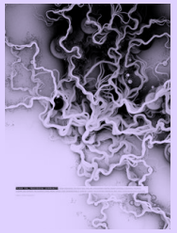
CON 225




CON 226

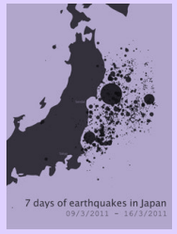


CON 227



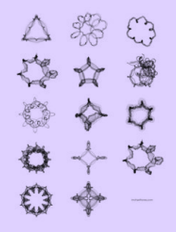
CON 228


CON 229




CON 230

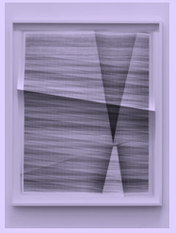
CON 231



CON 232


CON 233

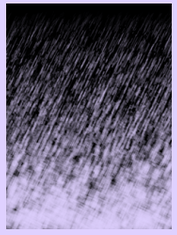
CON 234



CON 235

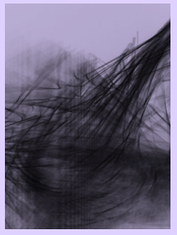
CON 236

CON 237

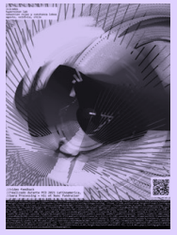


CON 238

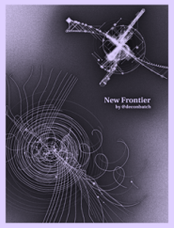

CON 239



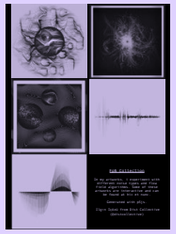
CON 240



CON 241



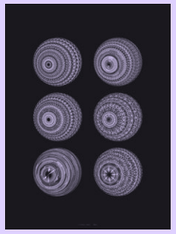

CON 242



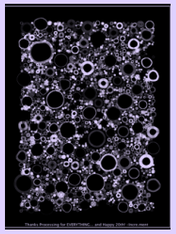
CON 243

CON 244


CON 245

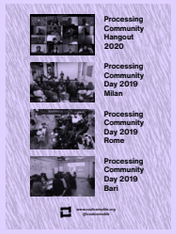


CON 246



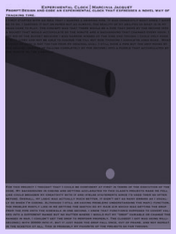
CON 247


CON 248




CON 249

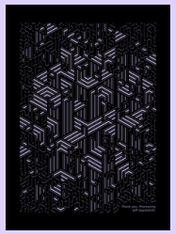

CON 250



CON 251


CON 252

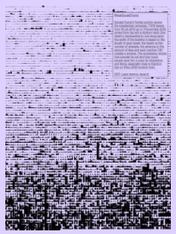

CON 253



CON 254

CON 255

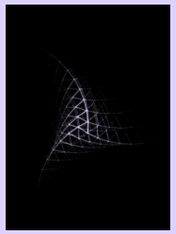

CON 256



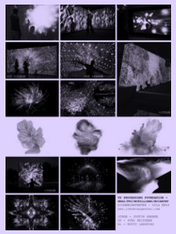
CON 257

CON 258


CON 259



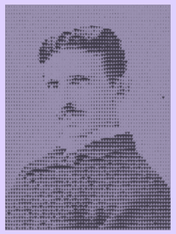
CON 260



CON 261

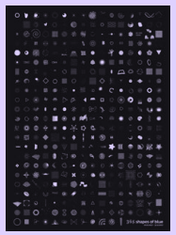

CON 262

CON 263



CON 264

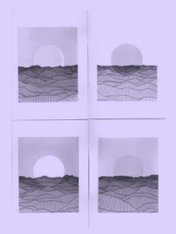
CON 265



CON 266

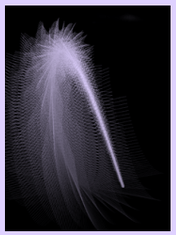

CON 267


CON 268



CON 269

CON 270





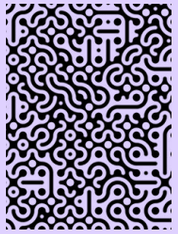
CON 271


CON 272

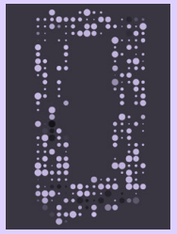

CON 273


CON 274

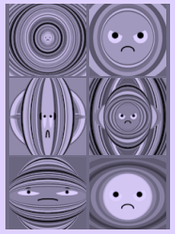
CON 275



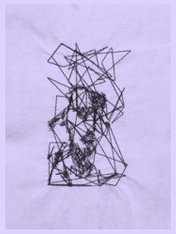
CON 276



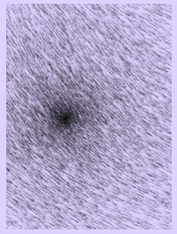
CON 277



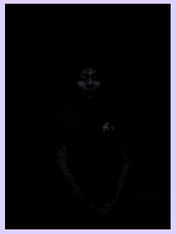
CON 278



CON 279

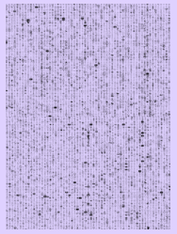


CON 280



CON 281


CON 282



CON 283

CON 284

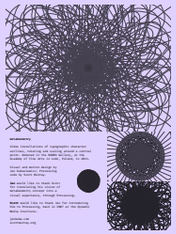

CON 285

CON 286


CON 287

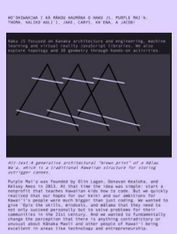

CON 288

CON 289

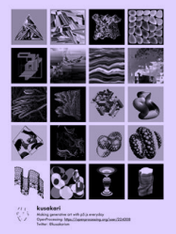


CON 290


CON 291



CON 292

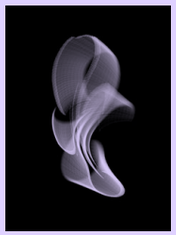


CON 293

CON 294

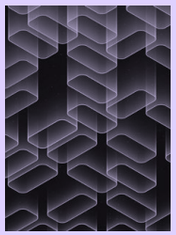
CON 295

CON 296



CON 297

CON 298



CON 299


CON 300

CON 301

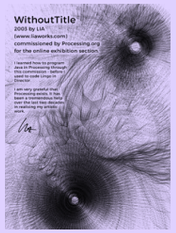

CON 302

CON 303

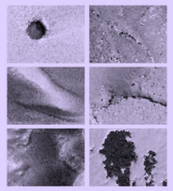
CON 304

CON 305

CON 306




CON 307



CON 308

CON 309

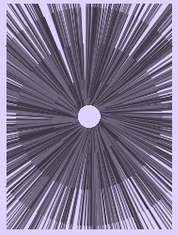

CON 310

CON 311

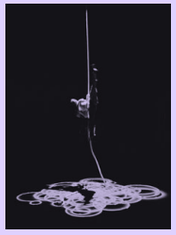

CON 312

CON 313

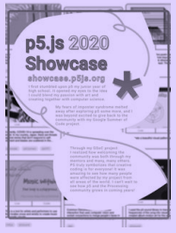
CON 314



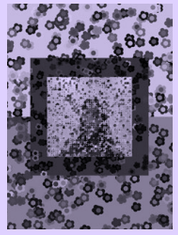
CON 315



CON 316




CON 317

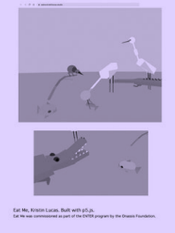


CON 318

CON 319


CON 320


CON 321



CON 322


CON 323

CON 324

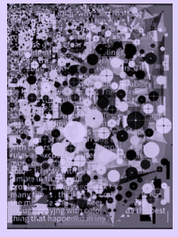

CON 325


CON 326

CON 327

CON 328

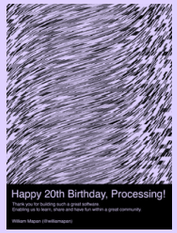

CON 329



CON 330



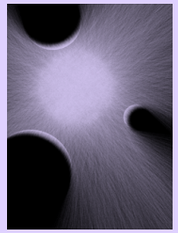
CON 331




CON 332

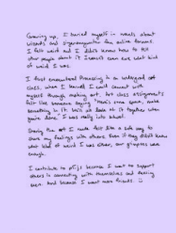
CON 333

CON 334

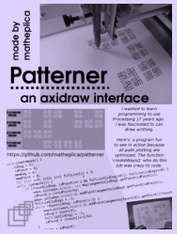


CON 335

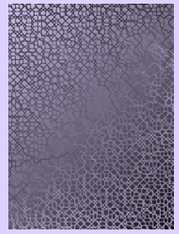
CON 336



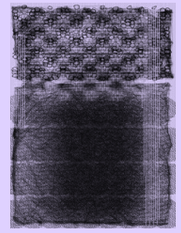
CON 337



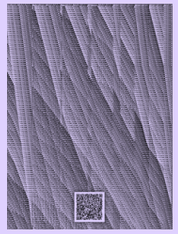

CON 338



CON 339

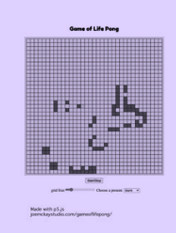


CON 340



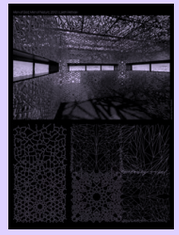
CON 341

CON 342




CON 343

CON 344



CON 345

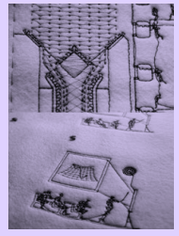
CON 346

CON 347

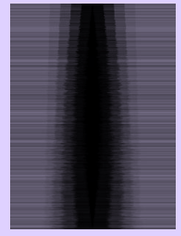
CON 348


CON 349

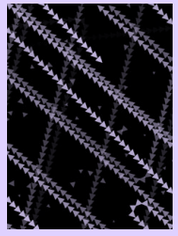
CON 350



CON 351



CON 352



CON 353

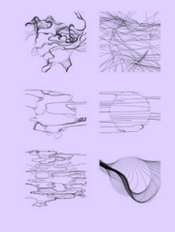

CON 354

CON 355

CON 356

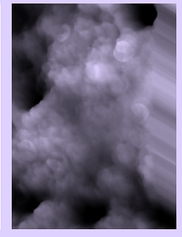
CON 357

CON 358

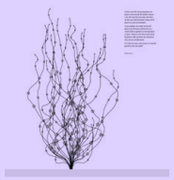


CON 359


CON 360

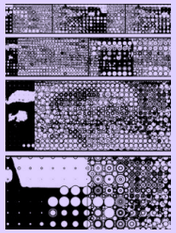


CON 361

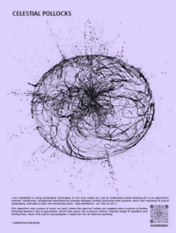


CON 362


CON 363



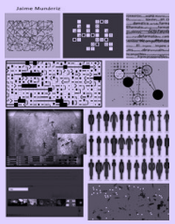
CON 364



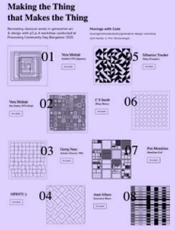

CON 365

CON 366

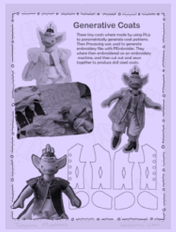

CON 367




CON 368

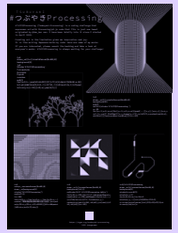



CON 369

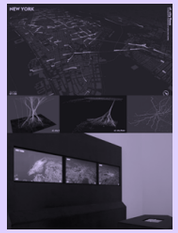



CON 370

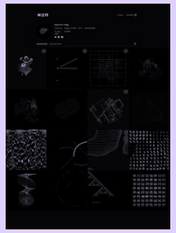

CON 371



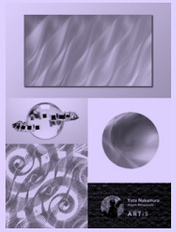

CON 372



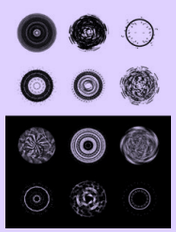

CON 373



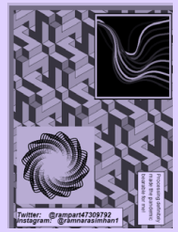
CON 374




CON 375



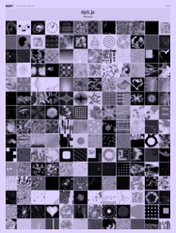
CON 376



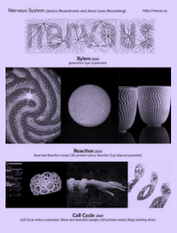

CON 377

CON 378


CON 379

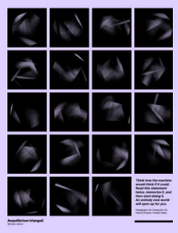



CON 380




CON 381

CON 382



CON 383

CON 384

CON 385

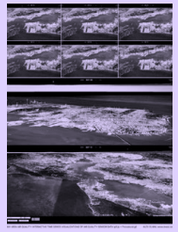

CON 386

CON 387


CON 388

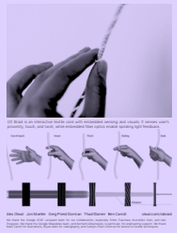

CON 389

CON 390



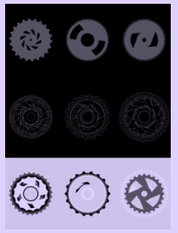


CON 391

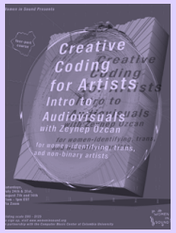


CON 392

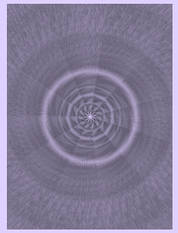
CON 393



CON 394



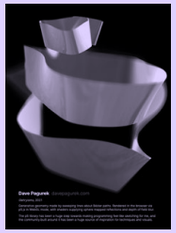

CON 395



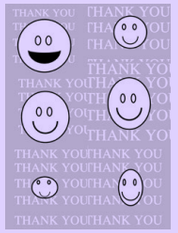
CON 396

CON 397


CON 398



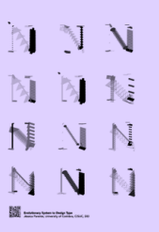

CON 399




CON 400

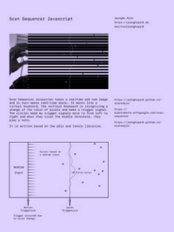

CON 401

CON 402



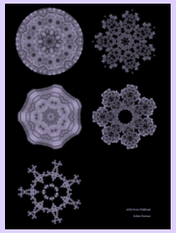
CON 403


CON 404

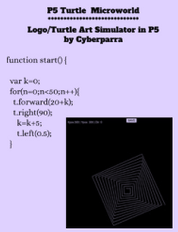



CON 405

CON 406



CON 407




CON 408


CON 409

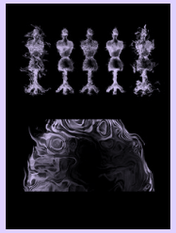
CON 410

CON 411


CON 412

CON 413

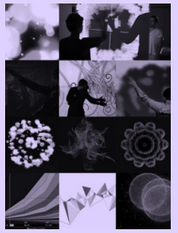

CON 414



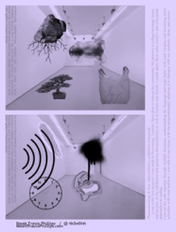
CON 415

CON 416


CON 417

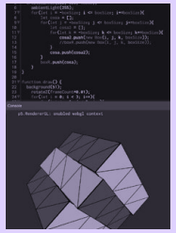


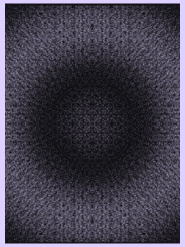
CON 418



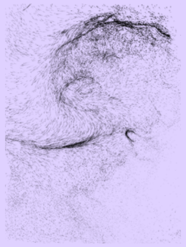
CON 419


CON 420



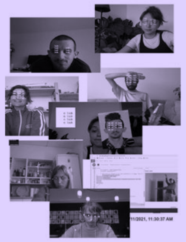
CON 421


CON 422

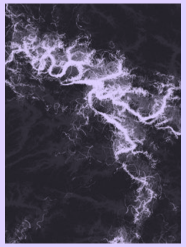

CON 423


CON 424


CON 425

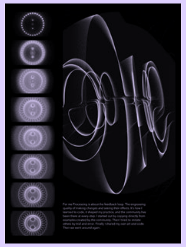

CON 426


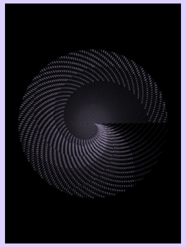
CON 427


CON 428


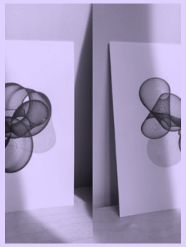
CON 429

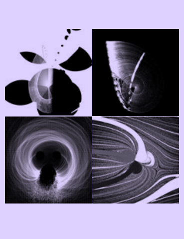

CON 430

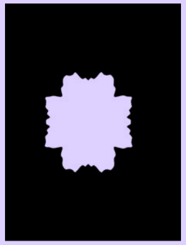

CON 431


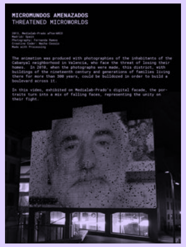
CON 432


CON 433

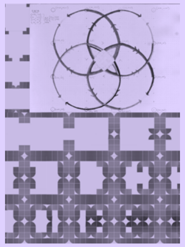

CON 434


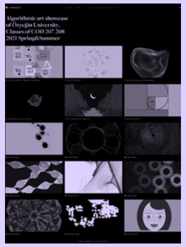
CON 435


CON 436


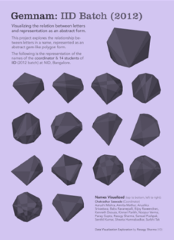
CON 437


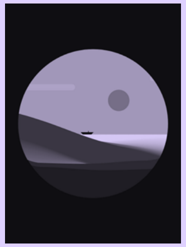
CON 438


CON 439


CON 440


CON 441


CON 442


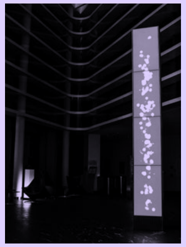
CON 443


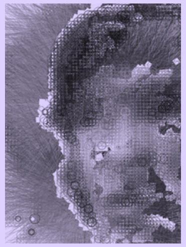
CON 444

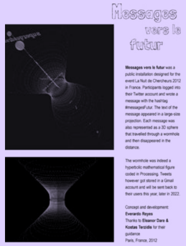

CON 445


CON 446

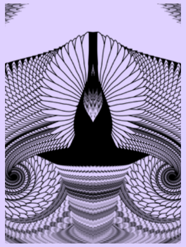

CON 447

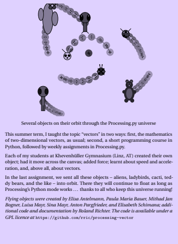

CON 448


CON 449


CON 450


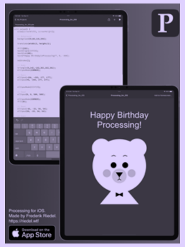
CON 451

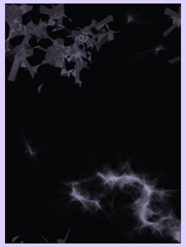

CON 452


CON 453


CON 454


CON 455


CON 456


CON 457


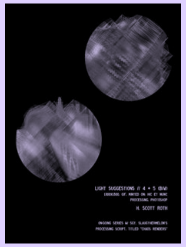
CON 458


CON 459

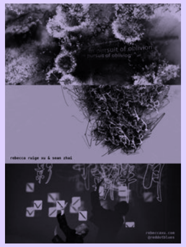

CON 460

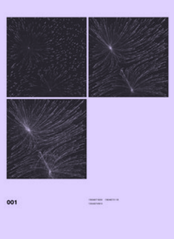

CON 461


CON 462


CON 463


CON 464

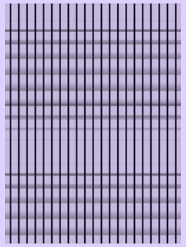

CON 465


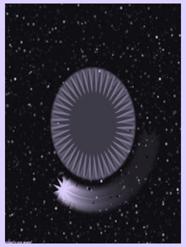
CON 466


CON 467

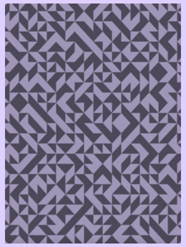

CON 468


CON 469


CON 470


CON 471


CON 472

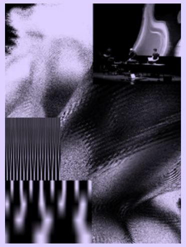

CON 473


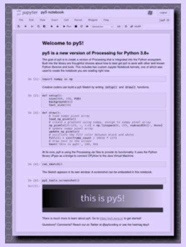
CON 474

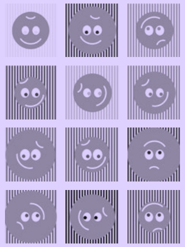

CON 475


CON 476


CON 477

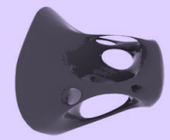

CON 478


CON 479


CON 480


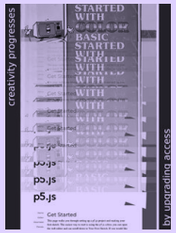
CON 481

CON 482



CON 483


CON 484



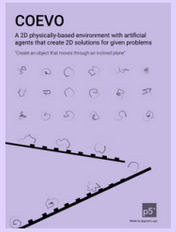

CON 485

CON 486

CON 487

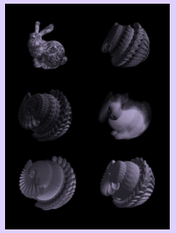

CON 488

CON 489




CON 490

CON 491

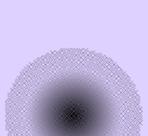


CON 492

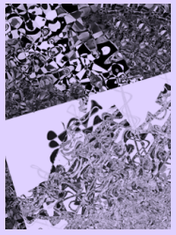
CON 493


CON 494

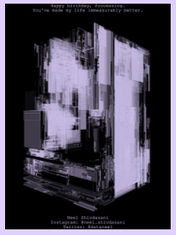
CON 495



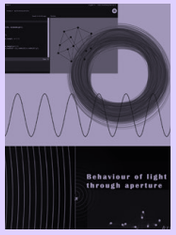
CON 496



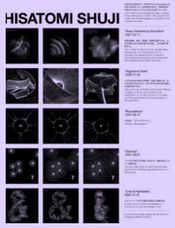
CON 497




CON 498

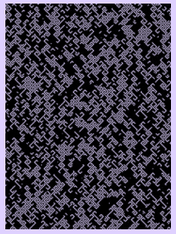



CON 499




CON 500


CON 501



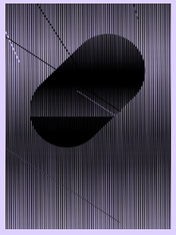
CON 502

CON 503


CON 504

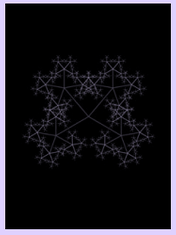

CON 505


CON 506



CON 507

CON 508



CON 509


CON 510



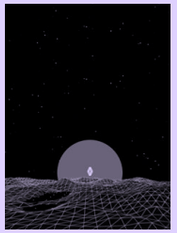
CON 511

CON 512


CON 513

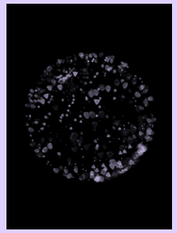
CON 514


CON 515



CON 516

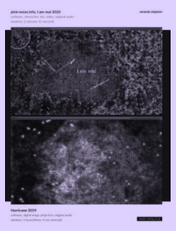
CON 517



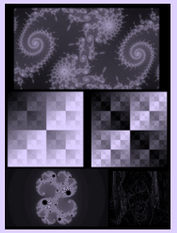
CON 518

CON 519

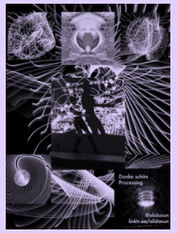
CON 520



CON 521



CON 522




CON 523

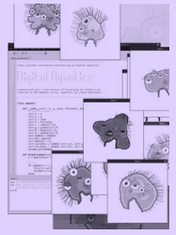
CON 524


CON 525


CON 526

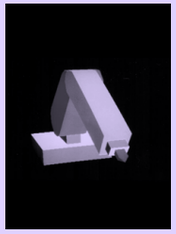

CON 527

CON 528

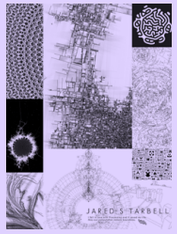


CON 529

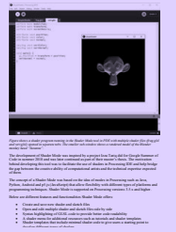

CON 530



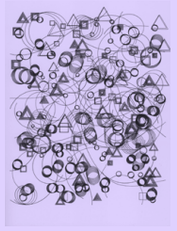
CON 531



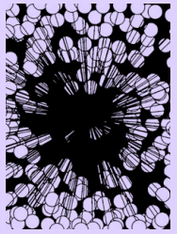

CON 532



CON 533



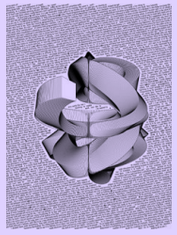
CON 534



CON 535

CON 536

CON 537



CON 538

CON 539

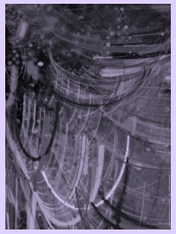
CON 540

CON 541

CON 542

CON 543

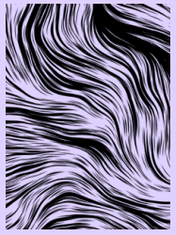

CON 544



CON 545


CON 546

CON 547



CON 548

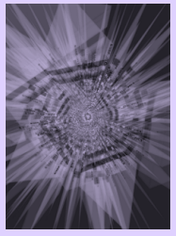
CON 549


CON 550

CON 551

CON 552

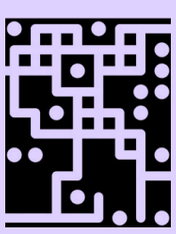

CON 553



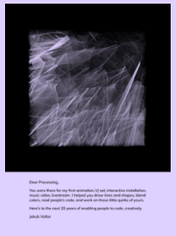
CON 554

CON 555

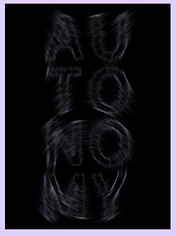

CON 556



CON 557



CON 558



CON 559

CON 560

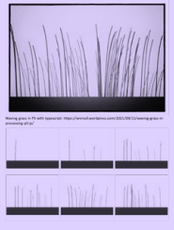

CON 561

CON 562

CON 563

CON 564

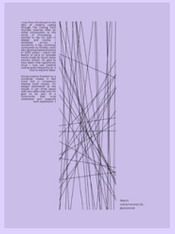
CON 565



CON 566


CON 567


CON 568

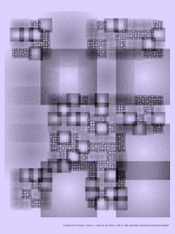


CON 569

CON 570

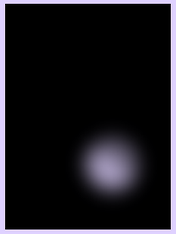


CON 571

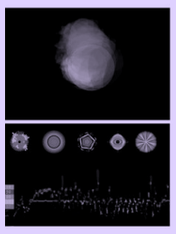


CON 572


CON 573

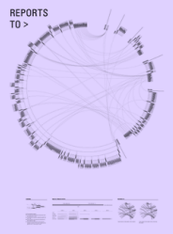


CON 574



CON 575

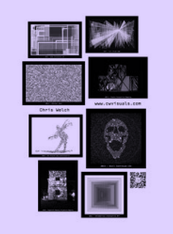
CON 576



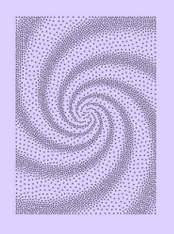

CON 577


CON 578

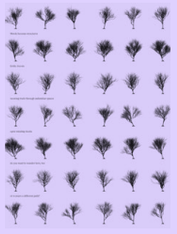

CON 579



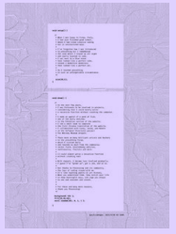

CON 580



CON 581

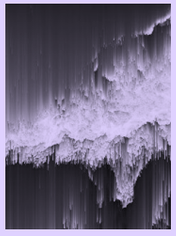


CON 582



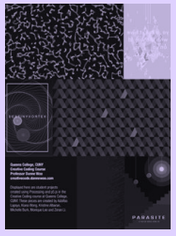
CON 583

CON 584

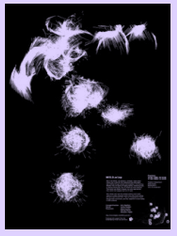


CON 585

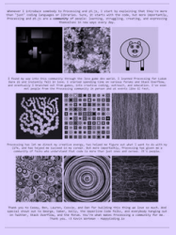

CON 586



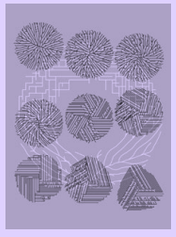

CON 587



CON 588

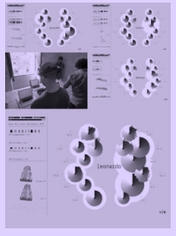


CON 589

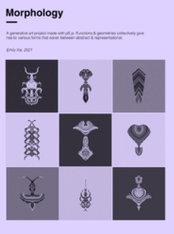


CON 590

CON 591



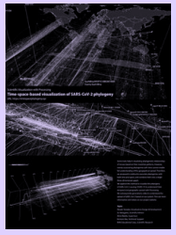
CON 592




CON 593

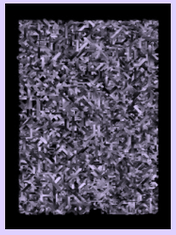

CON 594

CON 595



CON 596

CON 597



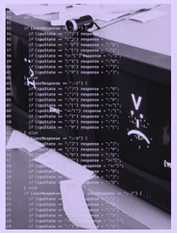
CON 598


CON 599

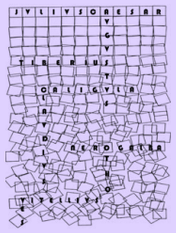

CON 600

CON 601


CON 602

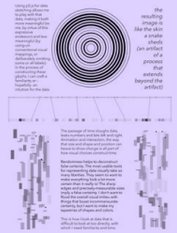


CON 603



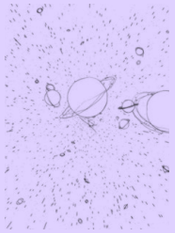
CON 604


CON 605



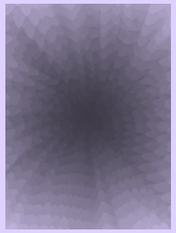
CON 606


CON 607

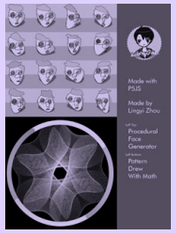


CON 608

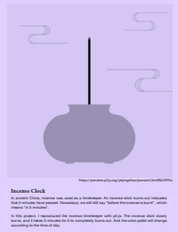
CON 609



CON 610



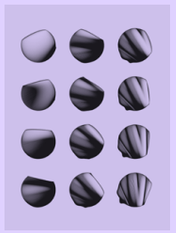

CON 611



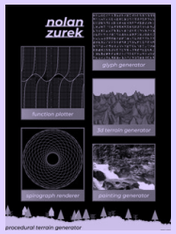

CON 612


CON 613

CON 614



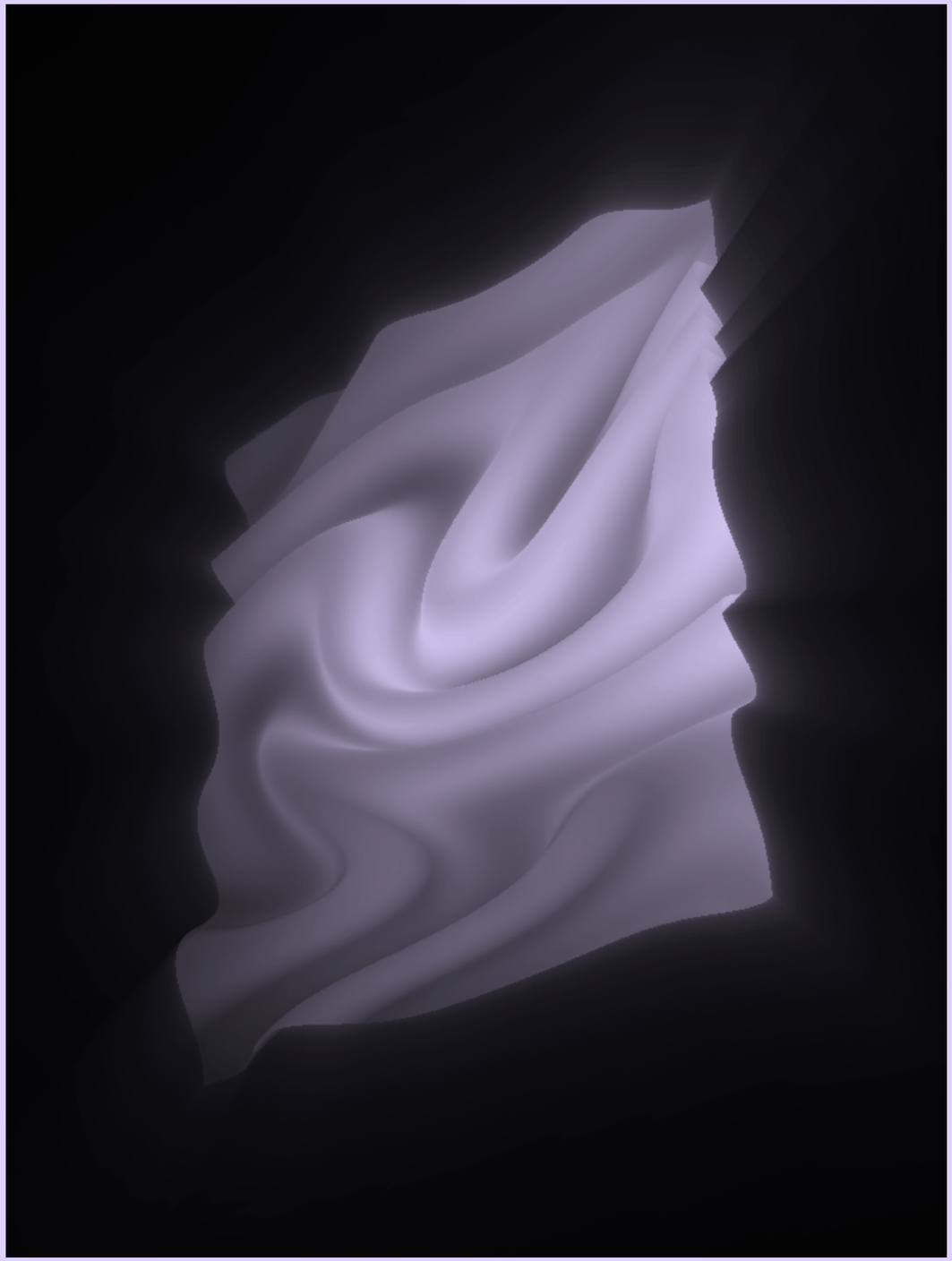
CON 615

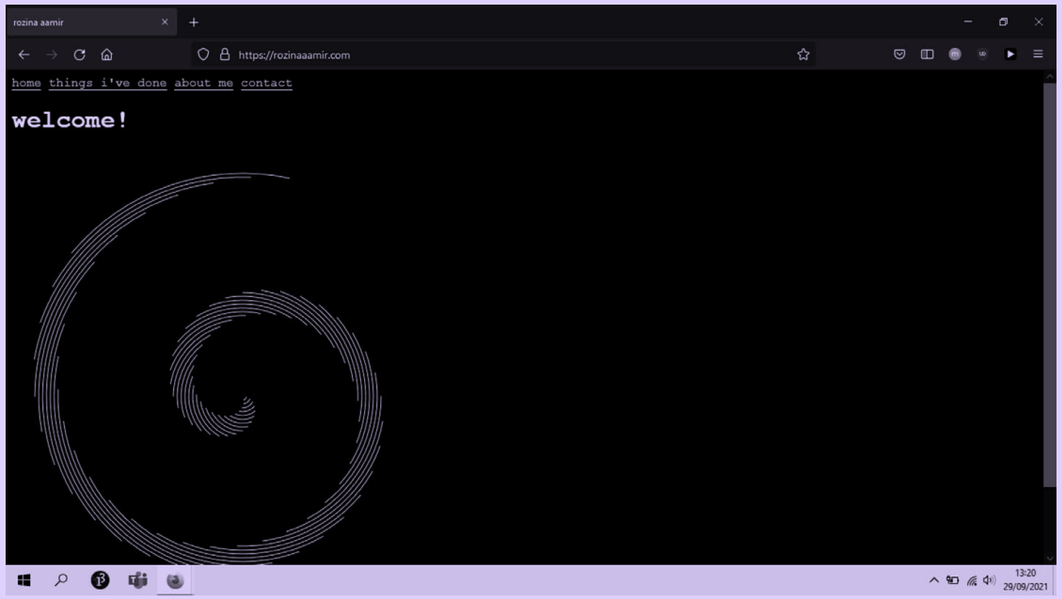


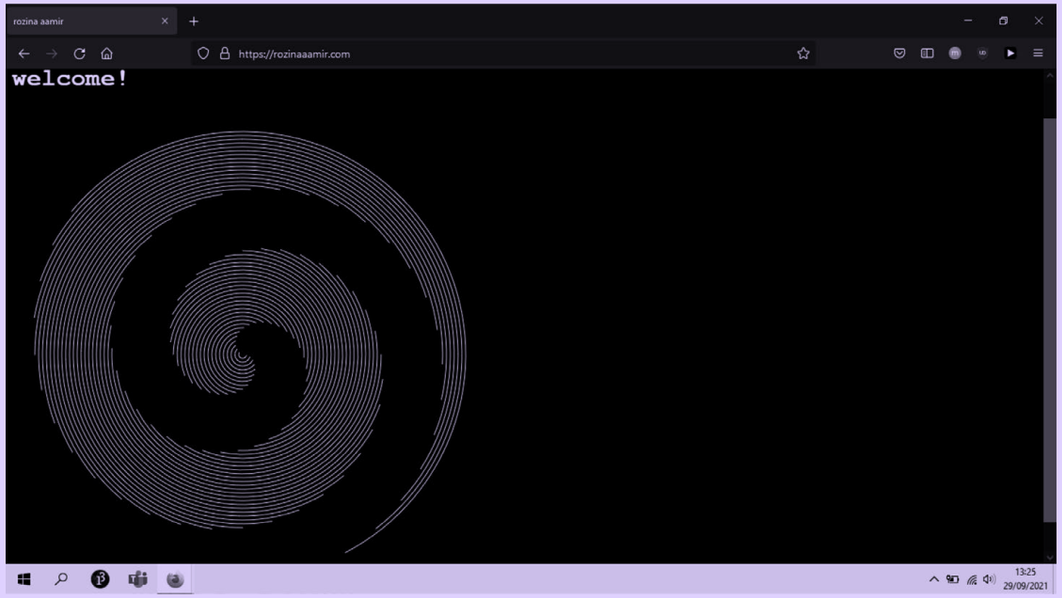

CON 616


meet creators
about Me You & U
< back
a lot of people assume
that I'm full Asian







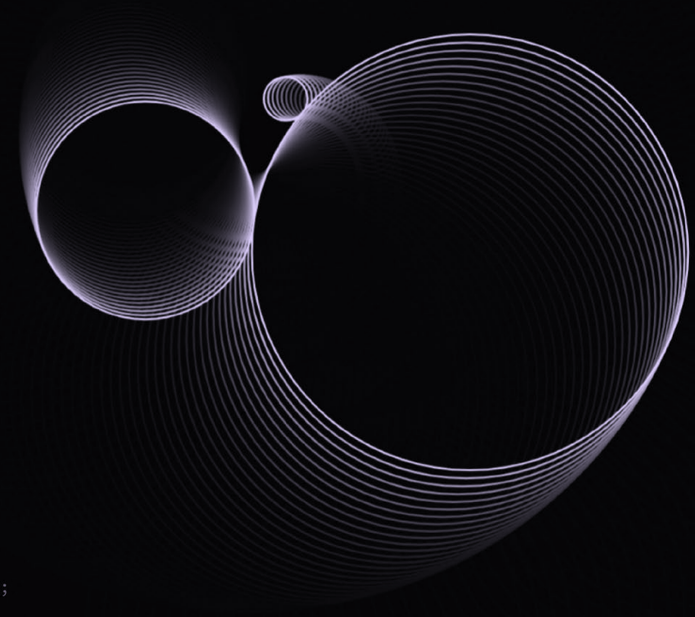






**spin spin spin**

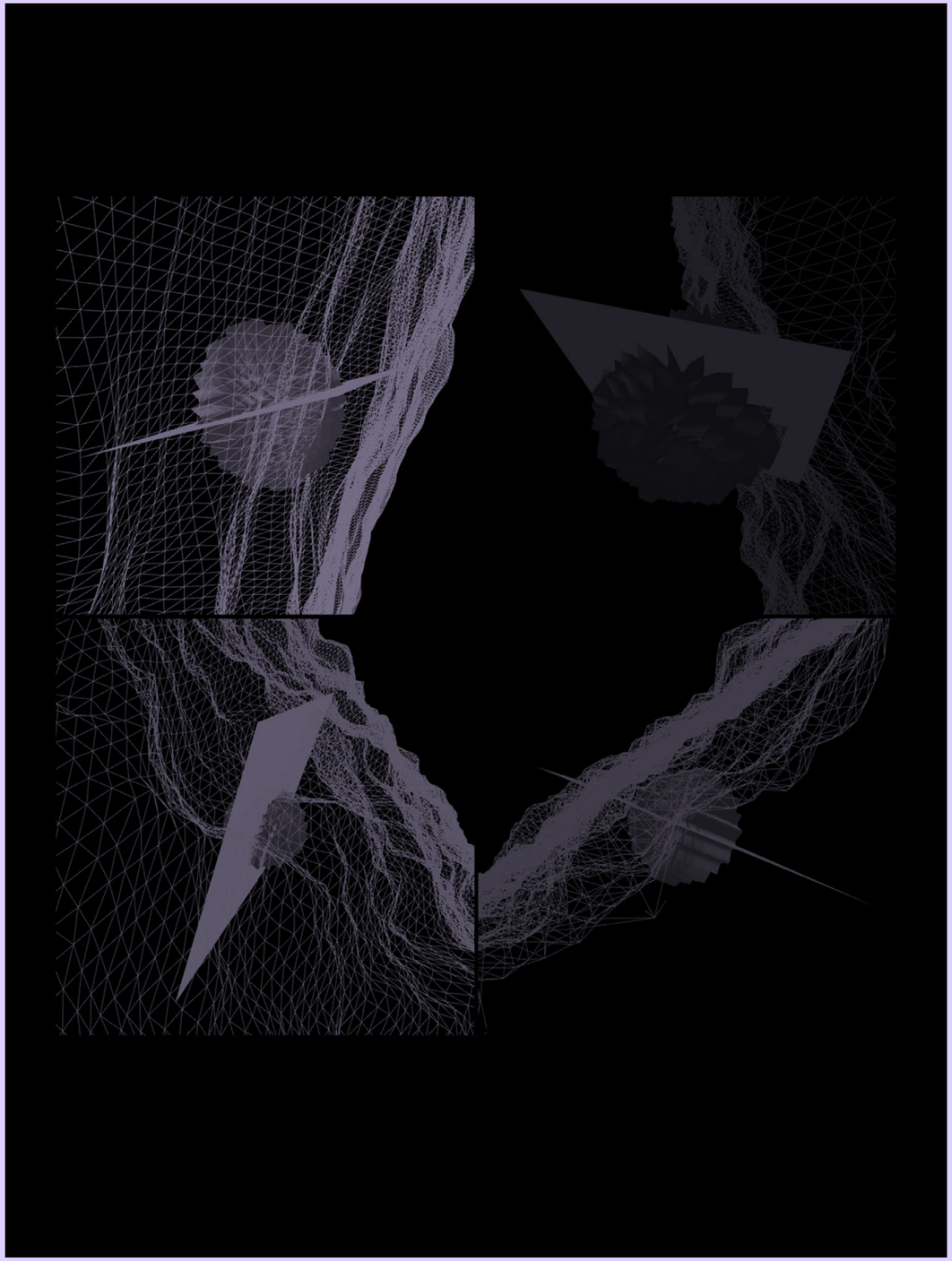
a circle, revolving around a circle, revolving around a circle
my first sketch

29th March, 2021
03:00am



```
let ang=0;

function setup() {
  createCanvas(windowWidth, windowHeight);
  noFill();
  stroke(255);
}

function draw() {
  background(0, 30);
  translate(width/2, height/2)
  let a=sin(ang);
  let b=cos(ang);
  let c=sin(ang*1.5);
  let d=cos(ang*1.5);
  let e=sin(ang*2);
  let f=cos(ang*2);
  circle(a*80, b*80, 100);
  circle(a*80+c*150, b*80+d*150, 200);
  circle(a*80+c*150+e*110, b*80+d*150+f*110, 20);
  ang+=0.025;
}
```

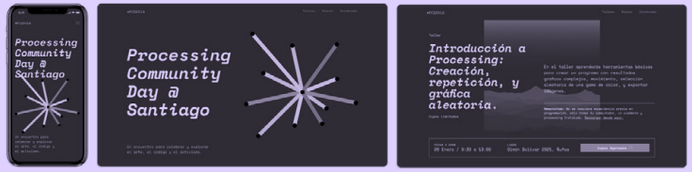
@aamirshaikh95

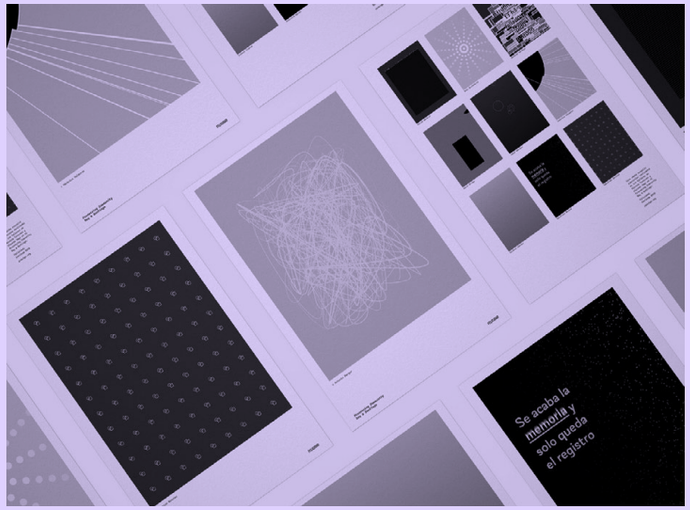




# Processing Community Day @ Santiago 2019

Under a festive, colorful and eye catching visual identity the chilean community gathered in a 1 day event divided into two parts: classes in the morning and talks in the afternoon. Students printed posters with their work and we distributed merchandizing that included code samples to test at home. Processing Community Day Santiago was an example that working with code is a collective work that is only possible if everyone participates.



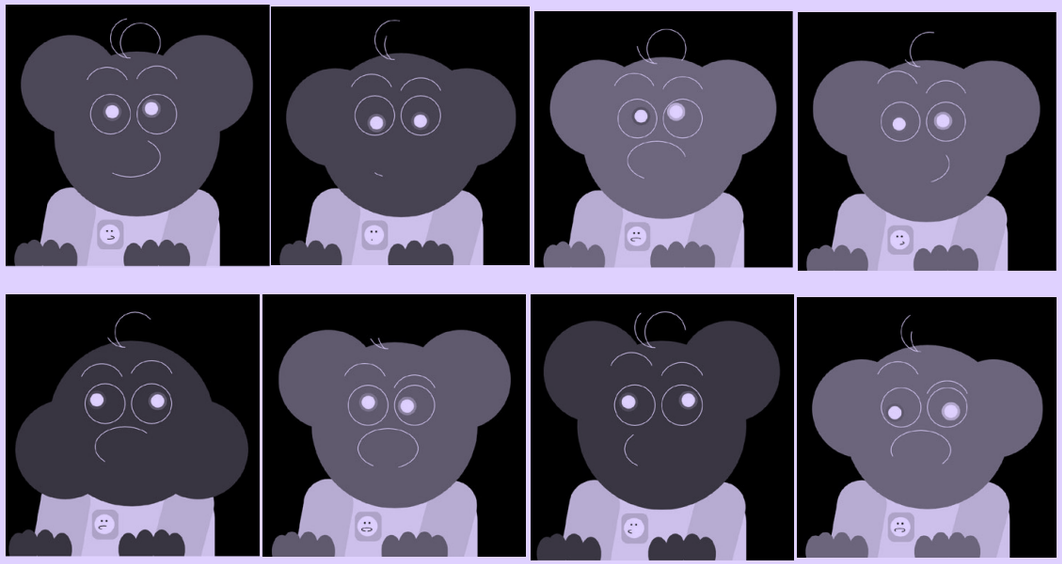




Organization
Alfredo Duarte

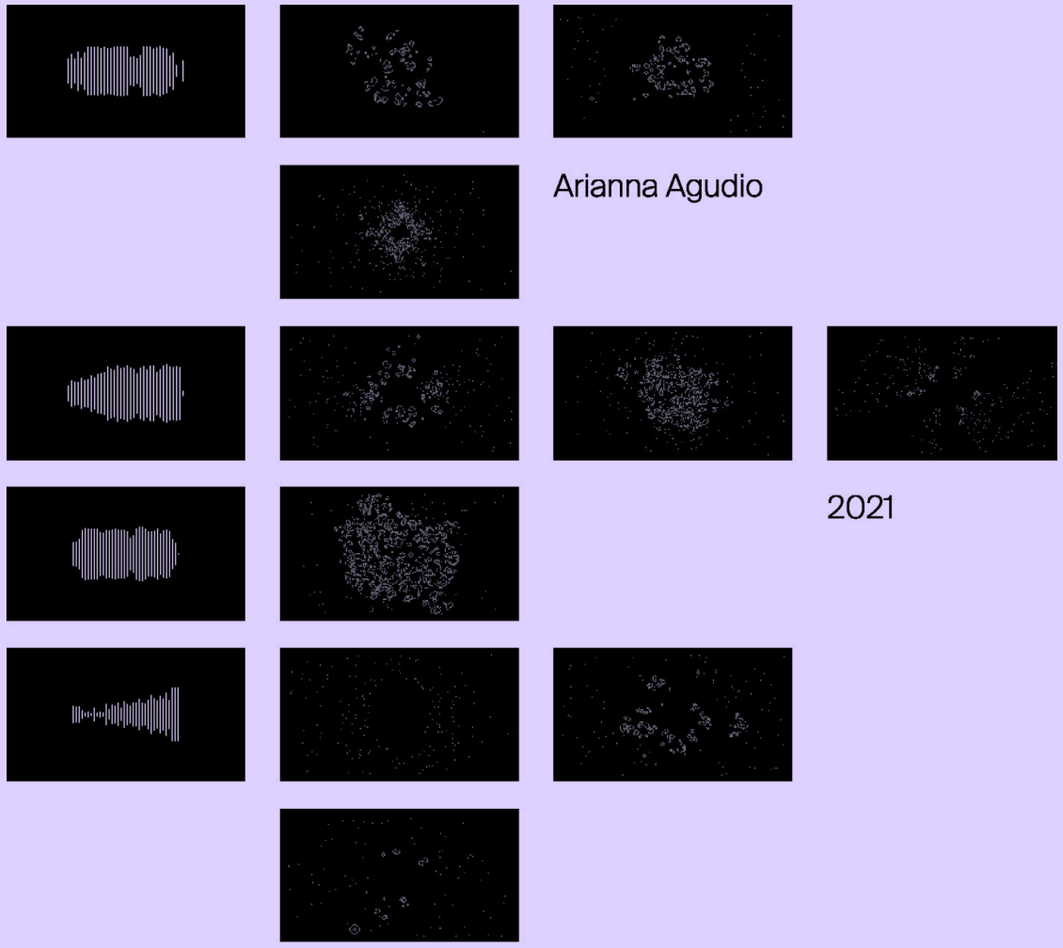
Production Team
Felicidad: Antonia Berger, Veronica Calderón, Christopher Cea, Tamara García, Victoria Gallardo, Isidora Jimenez, Consuelo Kehr, Josefa Labarca, Felipe Lozano, Jorge Navajas, Valentina Rubio.

Workshops
Nicolás Troncos, Ricardo Vega

Speakers
Roy McDonald, Juan De Magalhaes, Nicolás Troncoso, Pao olea, Javier Garay, Enrique Rivera





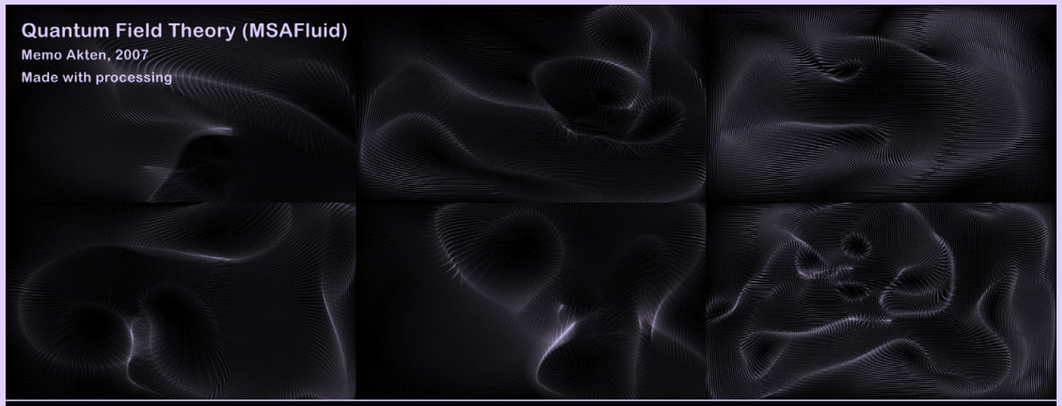


Arianna Agudio

2021

# The Sound of Life

To each soundscape its own generative universe.



## Quantum Field Theory (MSAFluid)

Memo Akten, 2007
Made with processing

## Nimiia Cétiï

**A collaboration between Jenna Sutela, Memo Akten, & Damien Henry from Google Arts & Culture, with support from Somerset House Studios, 2018. Inspired by experiments in interspecies communication and aspiring to connect with a world beyond our consciousness, Nimiia Cétiï documents the interactions between a neural network, audio recordings of early Martian language, & footage of the movements of extremophilic bacteria. Made with p5js**

16: cri déasi né ta téékiéé mo zé Mase a zénivi méré pa

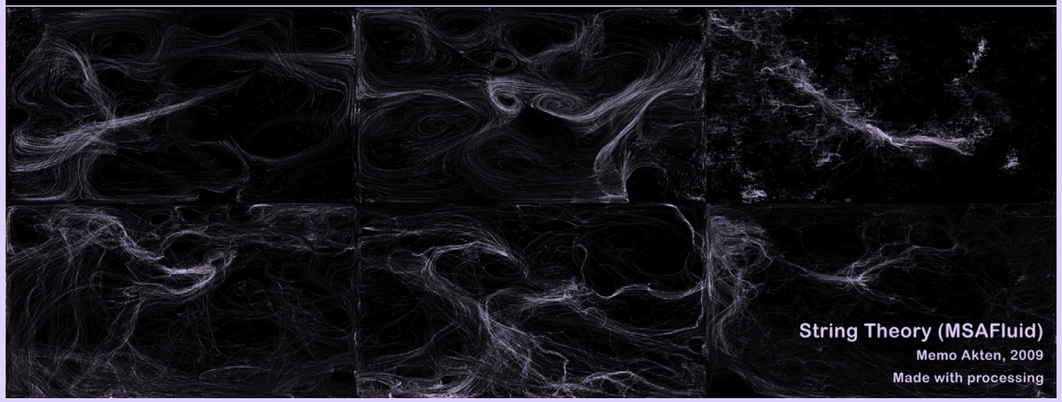
43: déni a dée si tenlé ti za délâtéchepou né tizi zamèz

67: tèscé nié mékétéchezé tisicapniétama dé céiâsitias

171: némi ké ta tar ni éa da ni fiêcé sizi sé ti néssé

231: miAr utouchi né sé sé zinée zazé misi dé ïa zé pé

316: tié vouitche mi grê si né ta ninastinèz mirénié



## String Theory (MSAFluid)

Memo Akten, 2009
Made with processing



Spaceship Edition of the

# ENGINE OF ENDLESS EXPLORATIONS



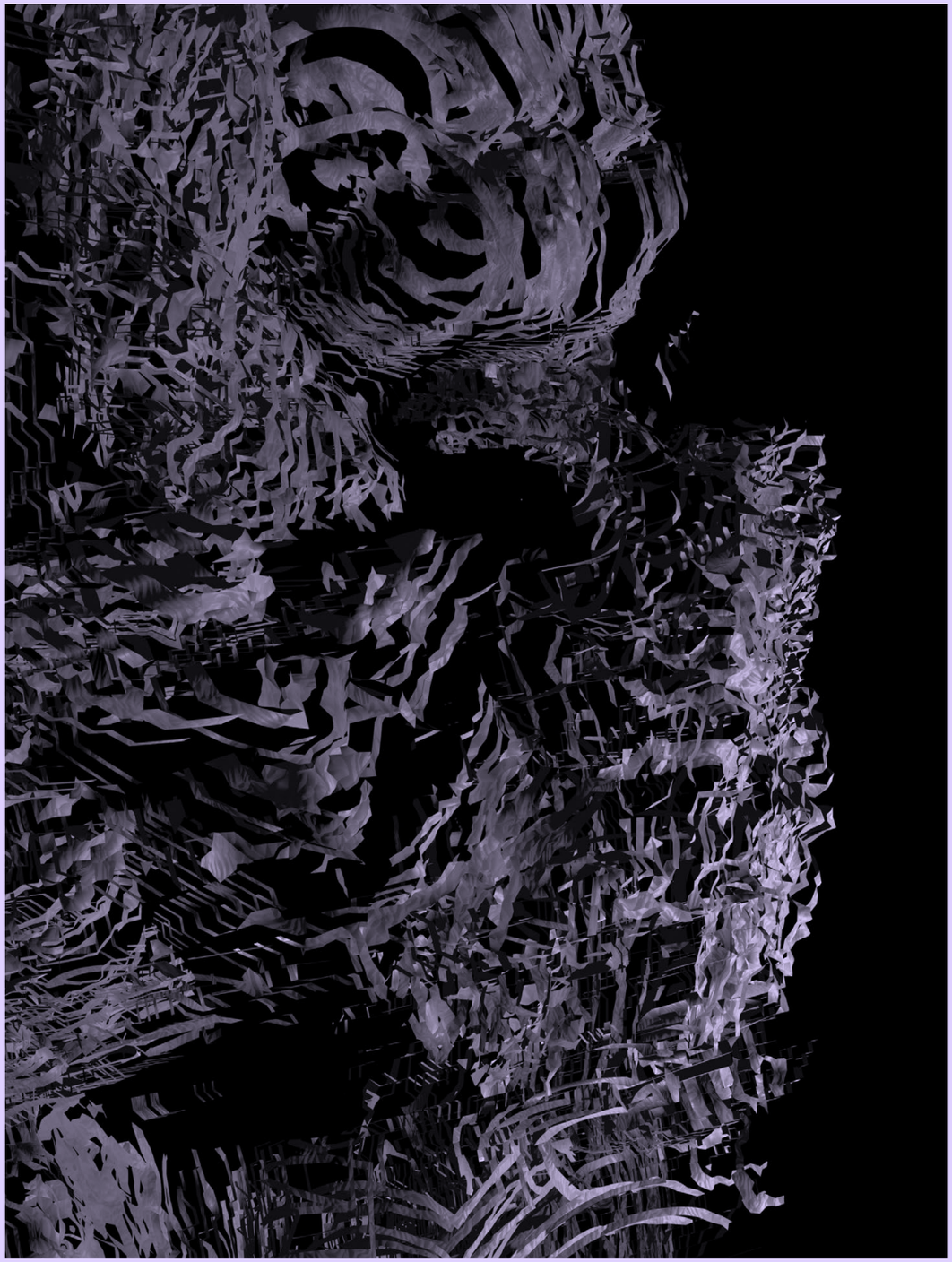
In the past decade generative design has become an emergent area of creative practice. Instead of designing individual objects, systems are programmed with rule-based controls to iterate over potential outcomes. These systems utilize random generation within fixed parameters to procedurally create endless explorations of the form they are designed to make.

This thesis investigates the potential of code as a tool for personal creative practice, specifically for generative graphic illustrations and form-making. This edition represents the variations of spaceships, made from geometric shapes, inspired by the space race illustrations of the 1950s and 60s. The iterative capacity of the procedural engine mirrors the human ambition that fueled that era's explorations.

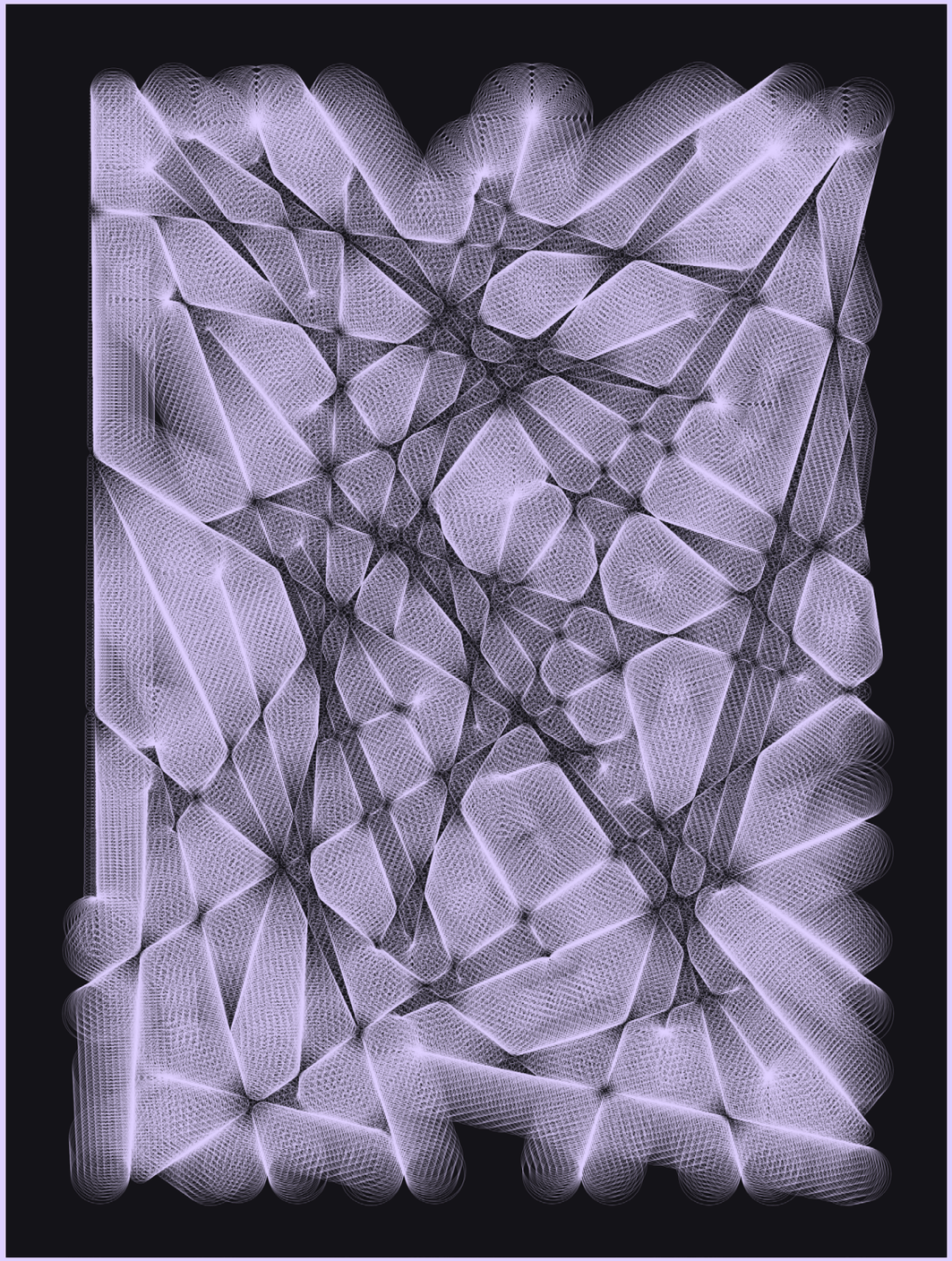
The intent for creating spaceships is a way for the viewer to playfully engage with the potential of code as a creative tool, and to spark their imagination and curiosity towards human space exploration.

*The structure of the code is based on a class system that stacks modular shapes to create endless iterations.*

*generative system built in Processing*
*project by @hindgalsaad*

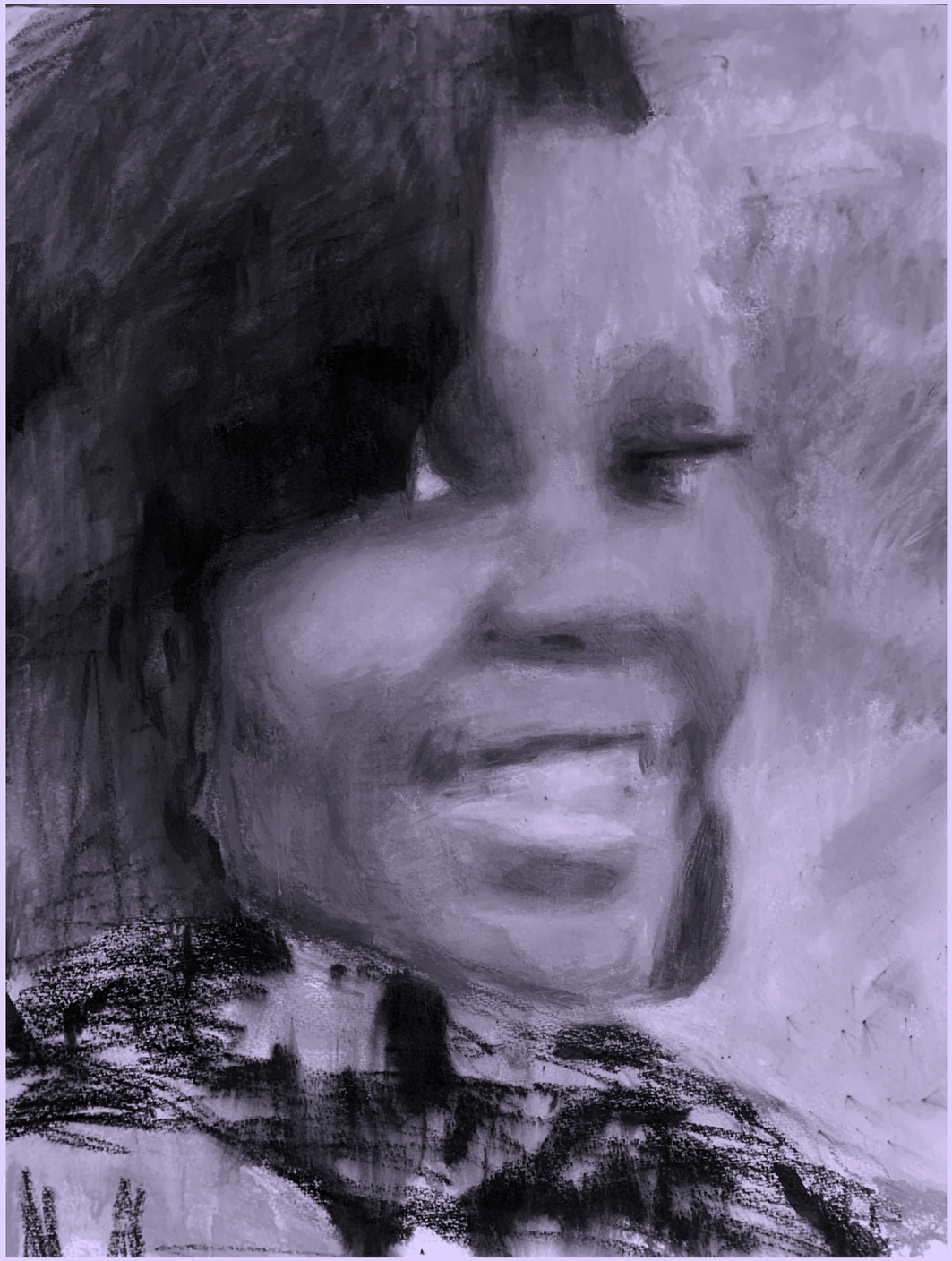







Algorat is a collective of these rats:
@carohermy
@poorly_documented
@crabbage_
@charstiles
We make rat based things on the web! These are some ecards
We made a markov valentine card generator using Rita.js and hallmark cards dataset
We love the Processing community and everything it stands for! Without p5, we wouldn't be able to do what we do! Thank you!
This is a 3D rat sweater creator!
This is a rat persona creator! We made a little gallery of all of rats submitted for this!
(If you make one tag @alg0rat on tw & IG)
RATMAKER
ALG0Rat Creator
backgrounds
bodies
eyes
reset!
randomize!
download!
@ameliawaltonn
@madeleinegrace_hall
@jasonphobic.xyz
@erinhklb
@taylor.niles
@samanthahernandz_
@tru3r0m4nc3
@bubblyvoid
@rennissence
These are all instagram handles
@ntschk
i@tscharlottewright
@_.peachslushie._



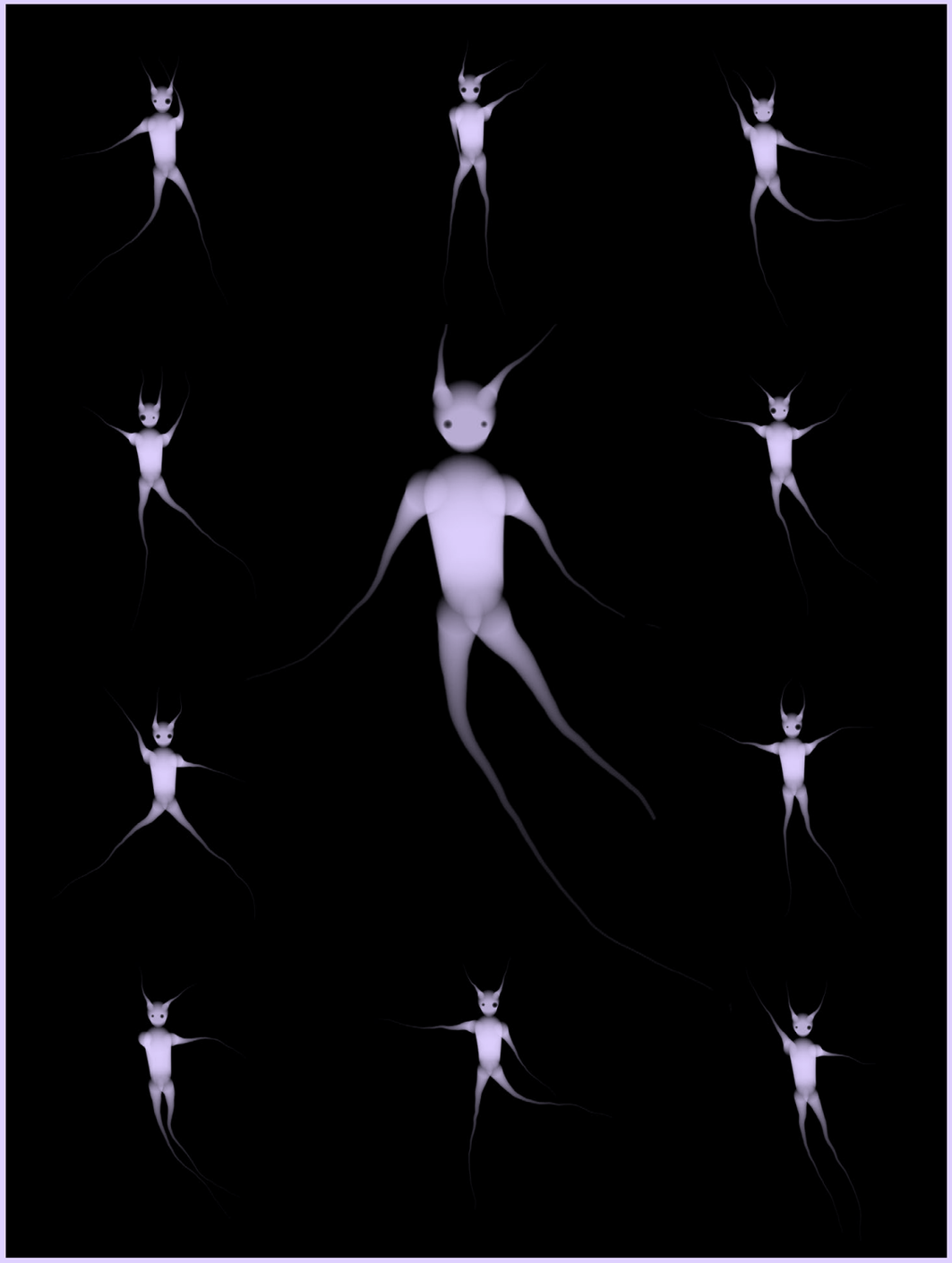


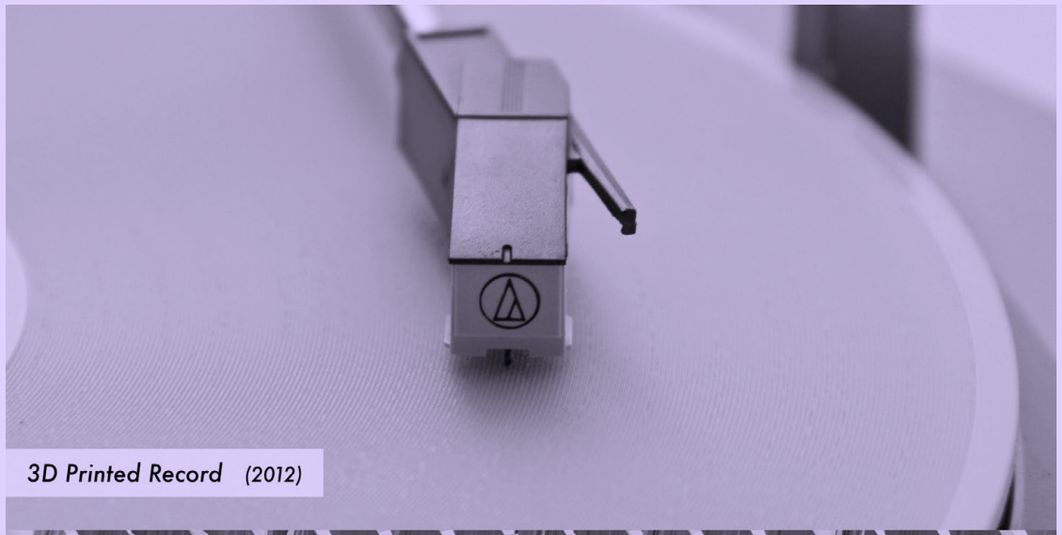
Lygia Pape

Livro do Tempo (1963)

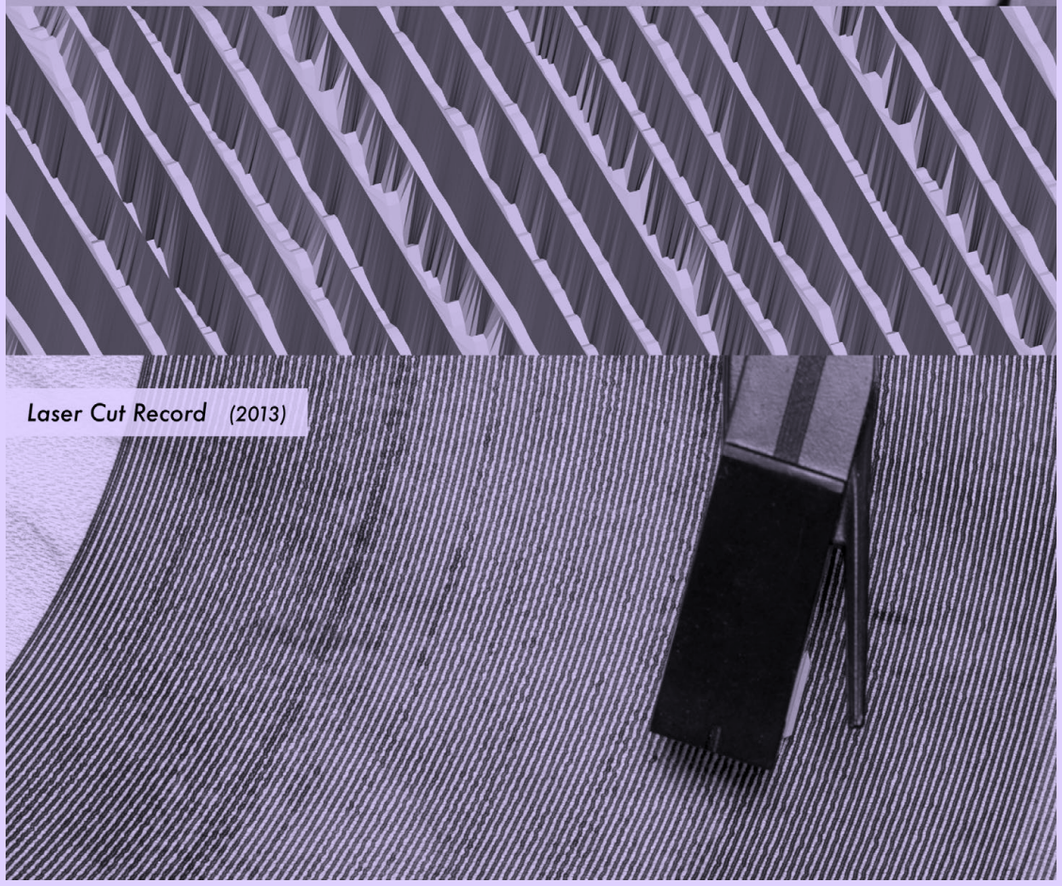
Recoded by Juan Alonso (@kokuma) for
CreativeCodingMadrid's recode project.
Code: https://tinyurl.com/livrorecoded





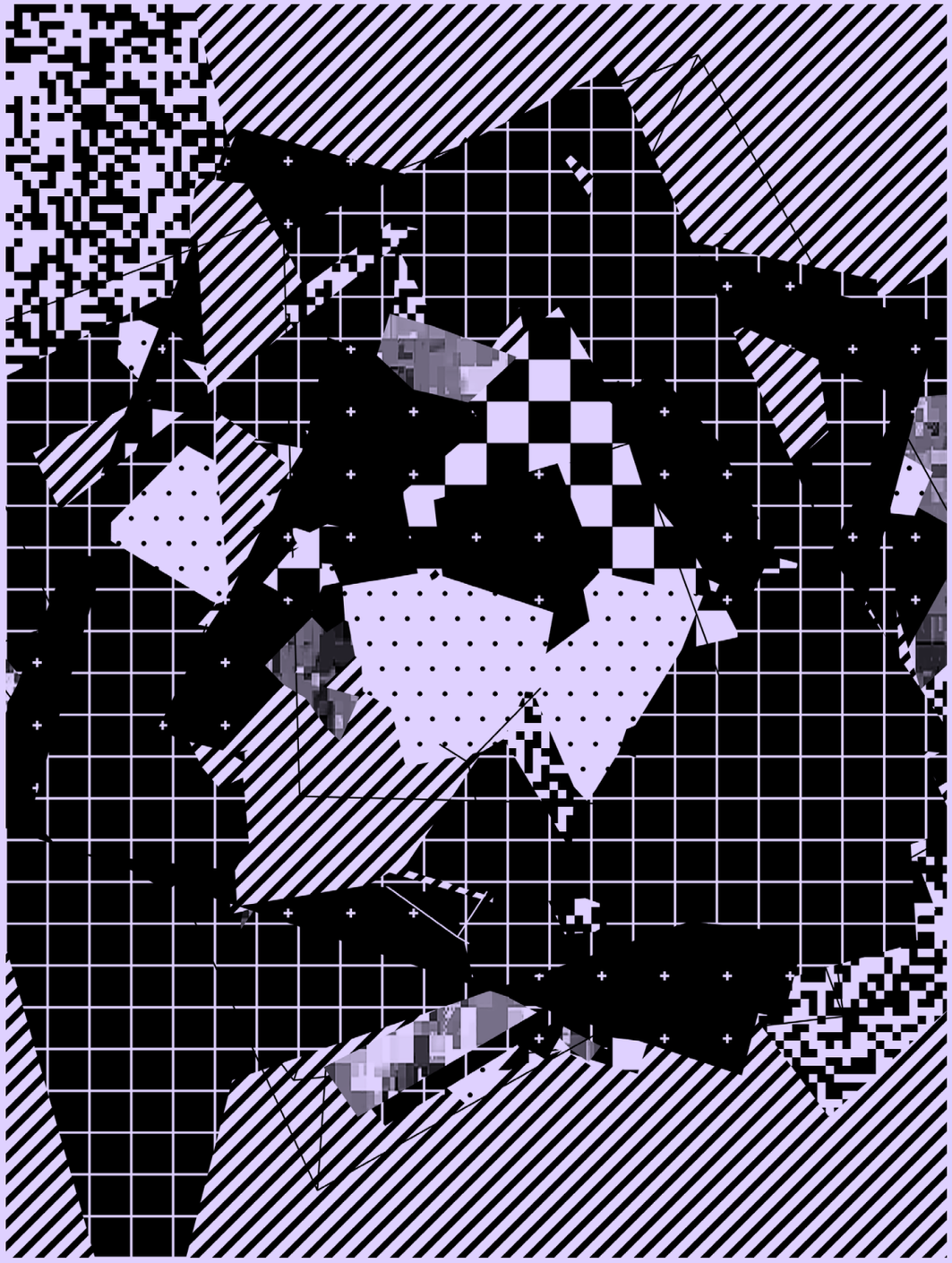


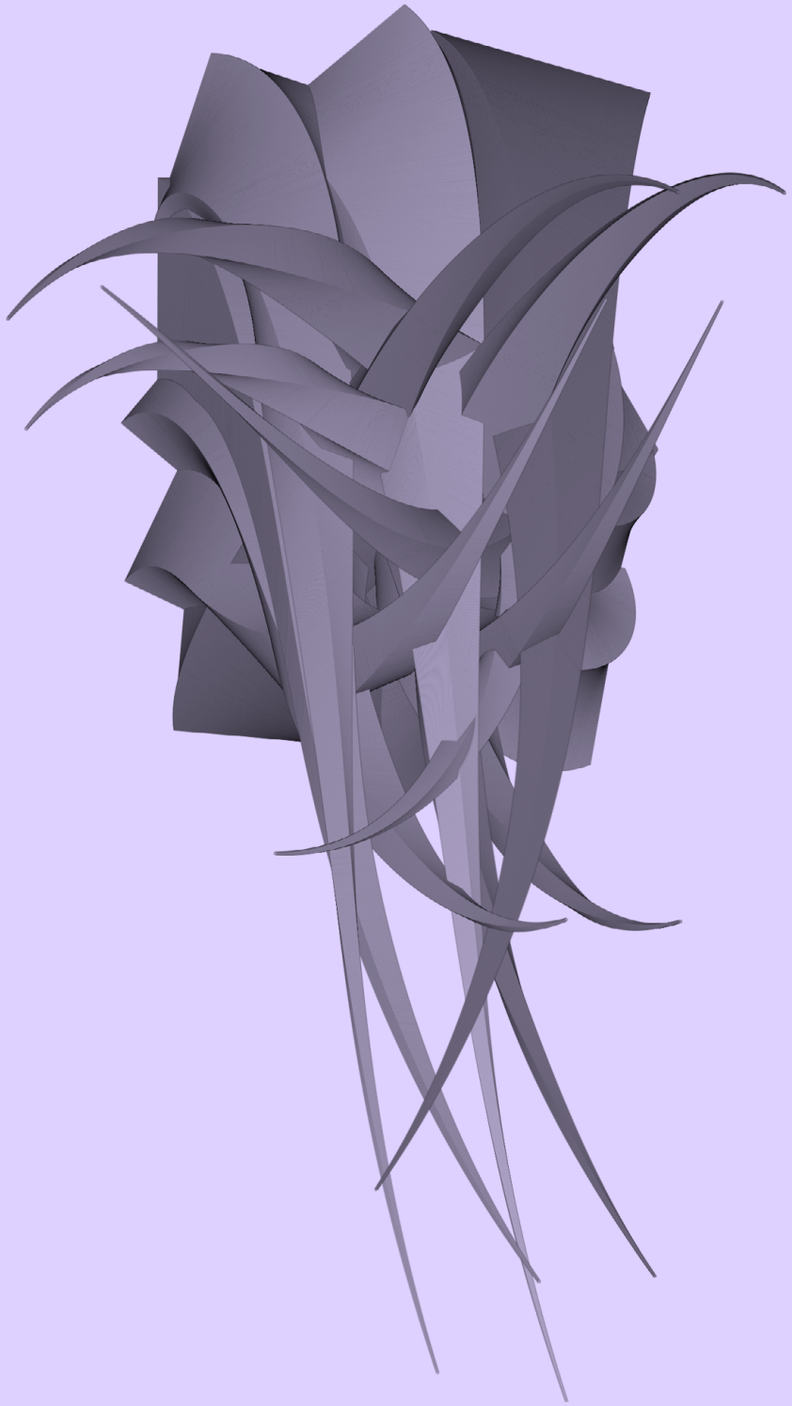
*3D Printed Record* (2012)



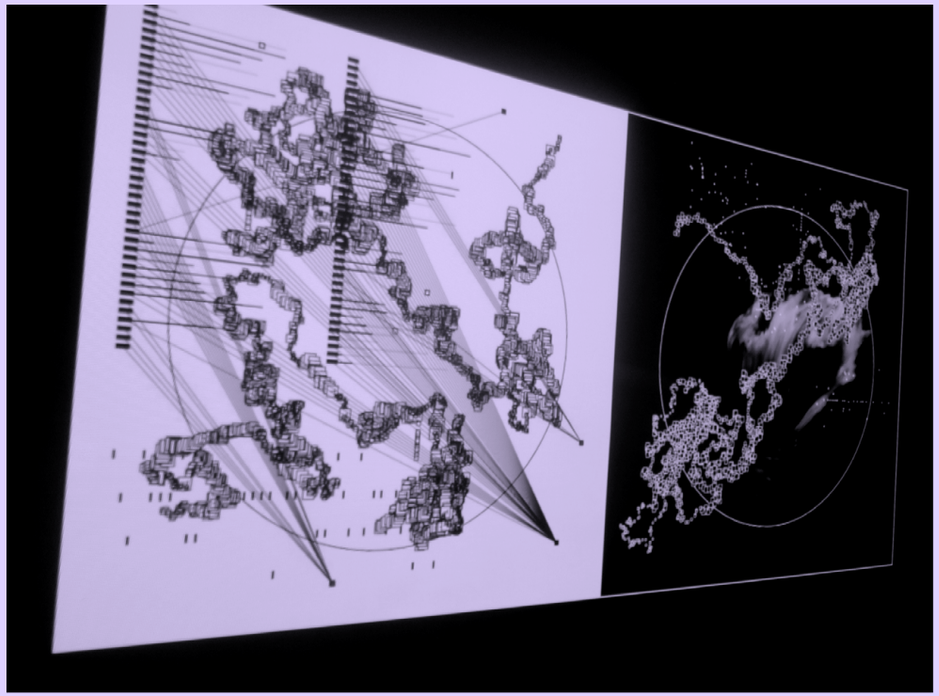
*Laser Cut Record* (2013)

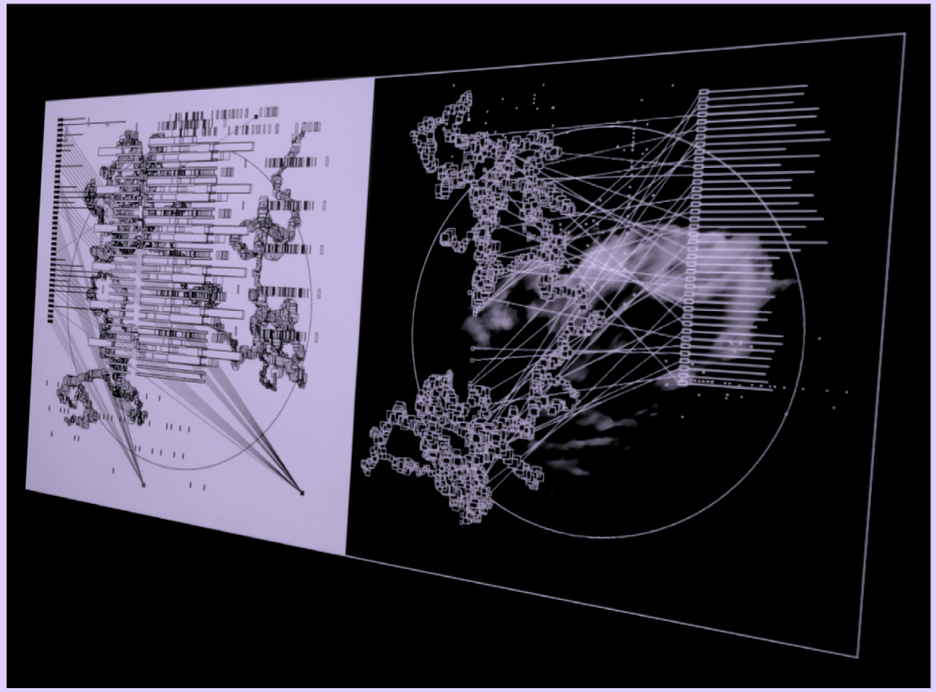


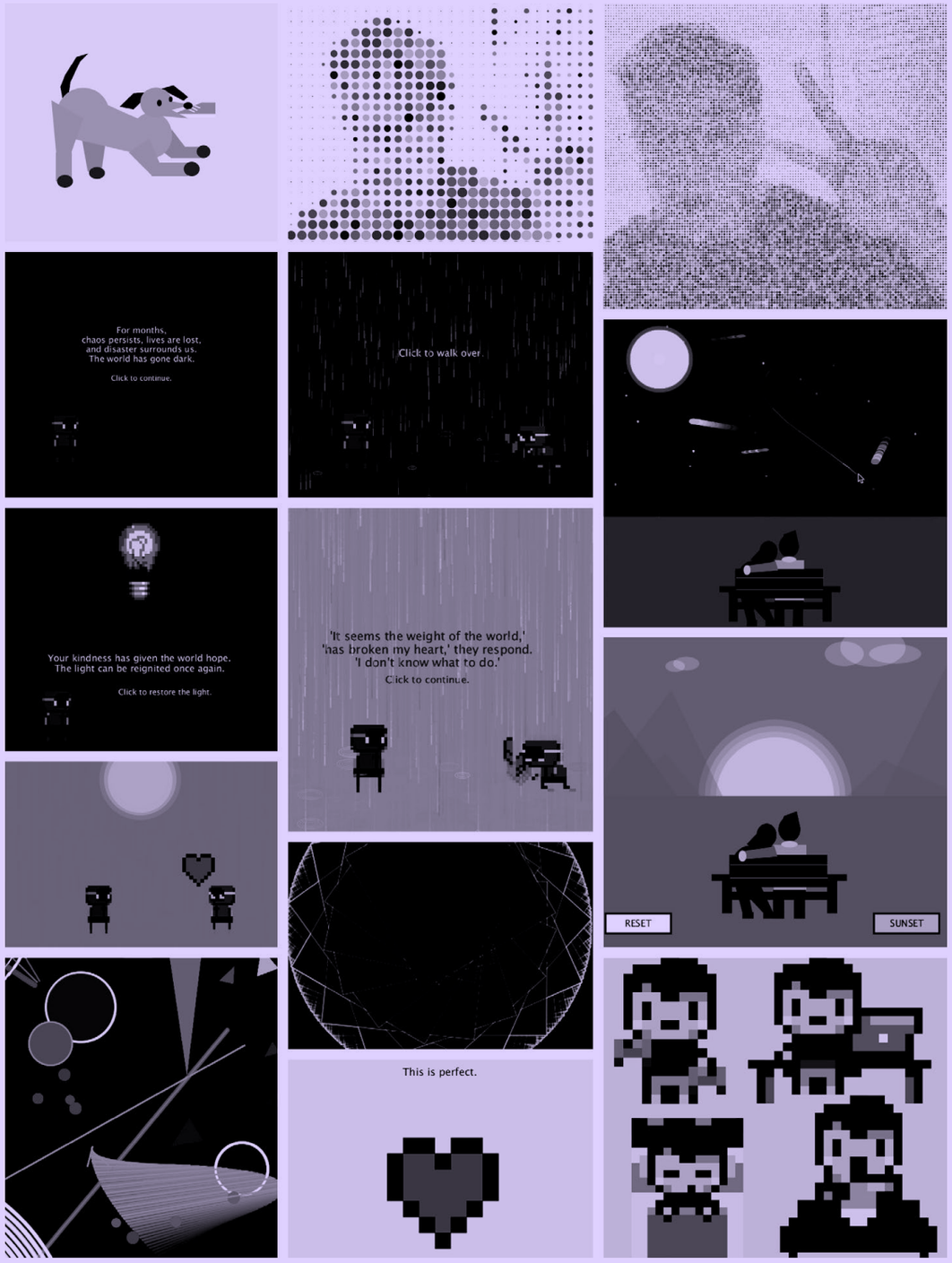




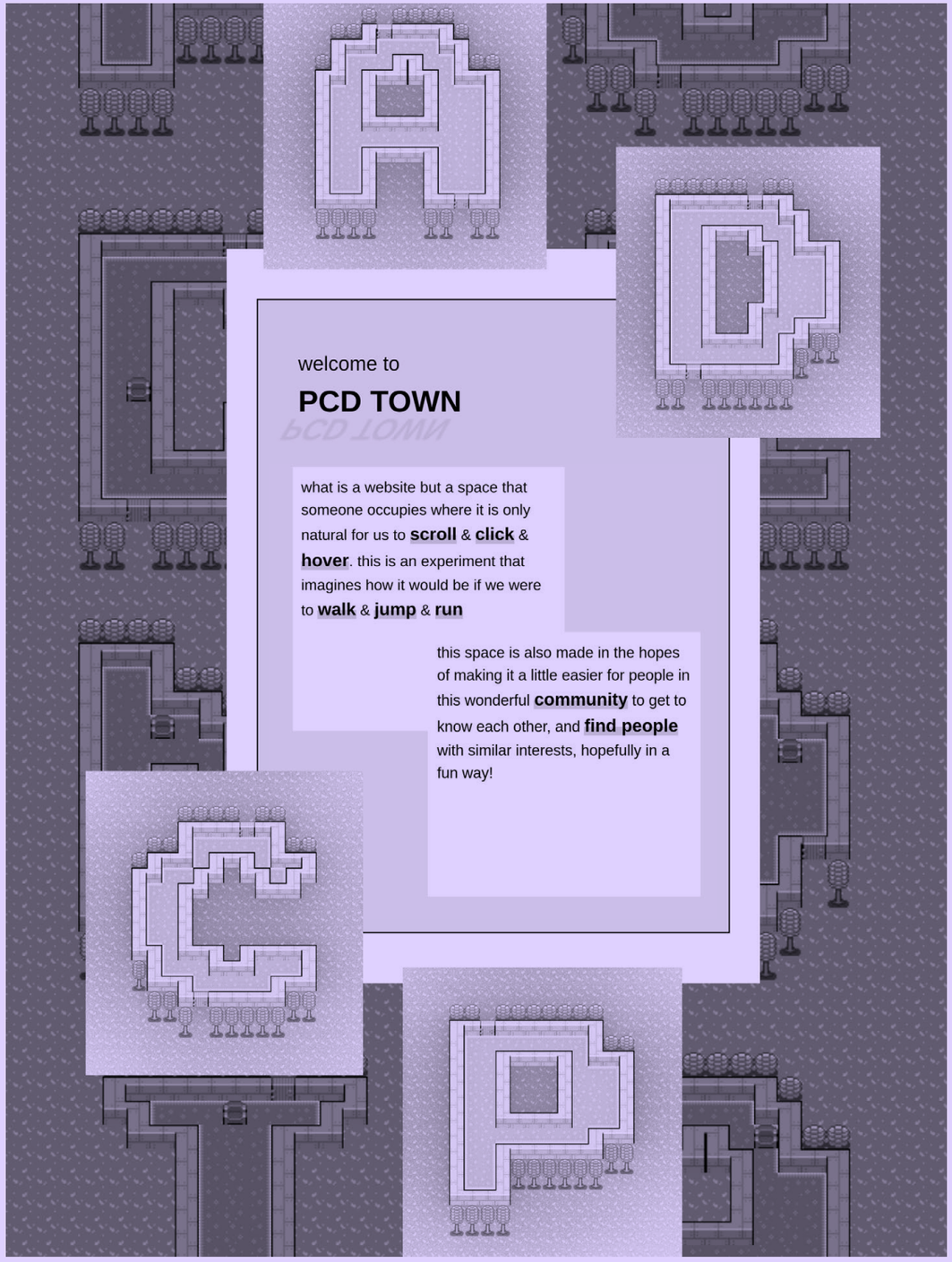







For months,
chaos persists, lives are lost,
and disaster surrounds us.
The world has gone dark.
Click to continue.
Click to walk over
Your kindness has given the world hope.
The light can be reignited once again.
Click to restore the light.
'It seems the weight of the world,'
'has broken my heart,' they respond.
'I don't know what to do.'
Click to continue.
RESET
SUNSET
This is perfect.

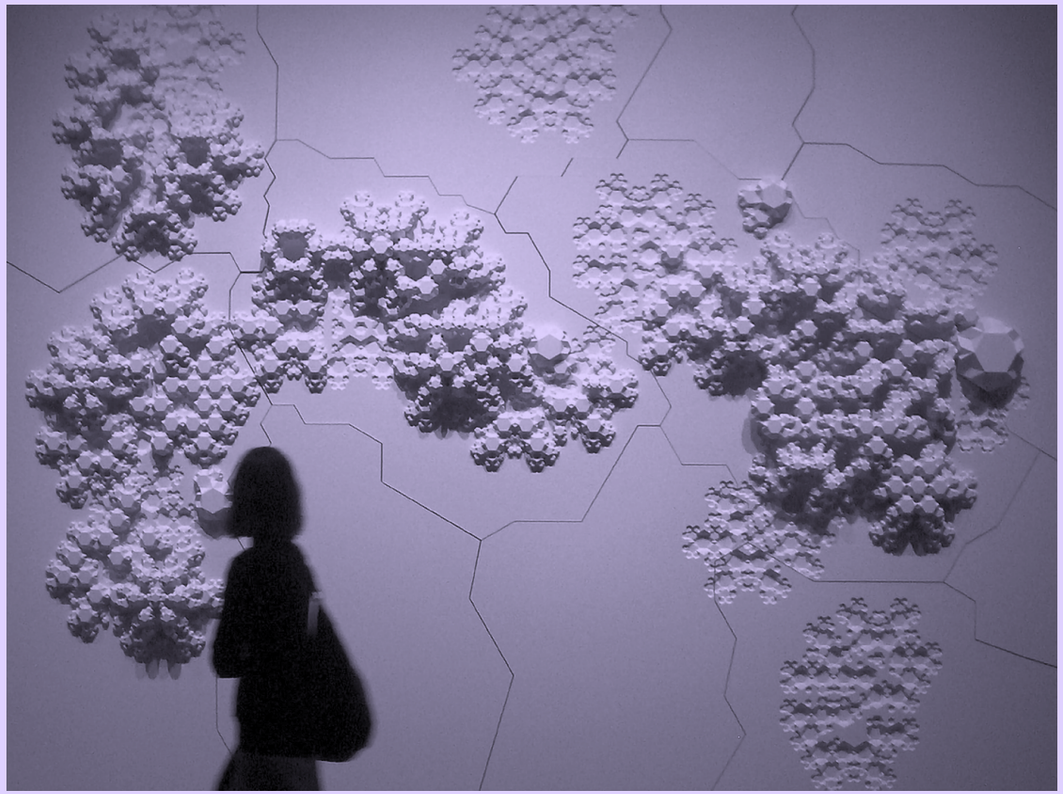



welcome to
PCD TOWN
what is a website but a space that someone occupies where it is only natural for us to scroll & click & hover. this is an experiment that imagines how it would be if we were to walk & jump & run
this space is also made in the hopes of making it a little easier for people in this wonderful community to get to know each other, and find people with similar interests, hopefully in a fun way!


```
var confidence = [];
var currentConfidence;

//add affirmations to the confidence array
  confidence.push(name + ", your art is good!")
  confidence.push(name + " deserves a million twitter followers!")
  confidence.push("oh " + name + ", you're the best!")
  confidence.push(name + ", you're doing the right thing")
```

```
confidence.push(name + ", I believe in you.")
confidence.push(name + ", YES!")
confidence.push(name + ", you're brilliant!")
confidence.push(name + ", you're perfect!")
confidence.push(name + ", your work should be in the museum!")
confidence.push(name + ", listen to me")
confidence.push(name + ", confidence now.")
```

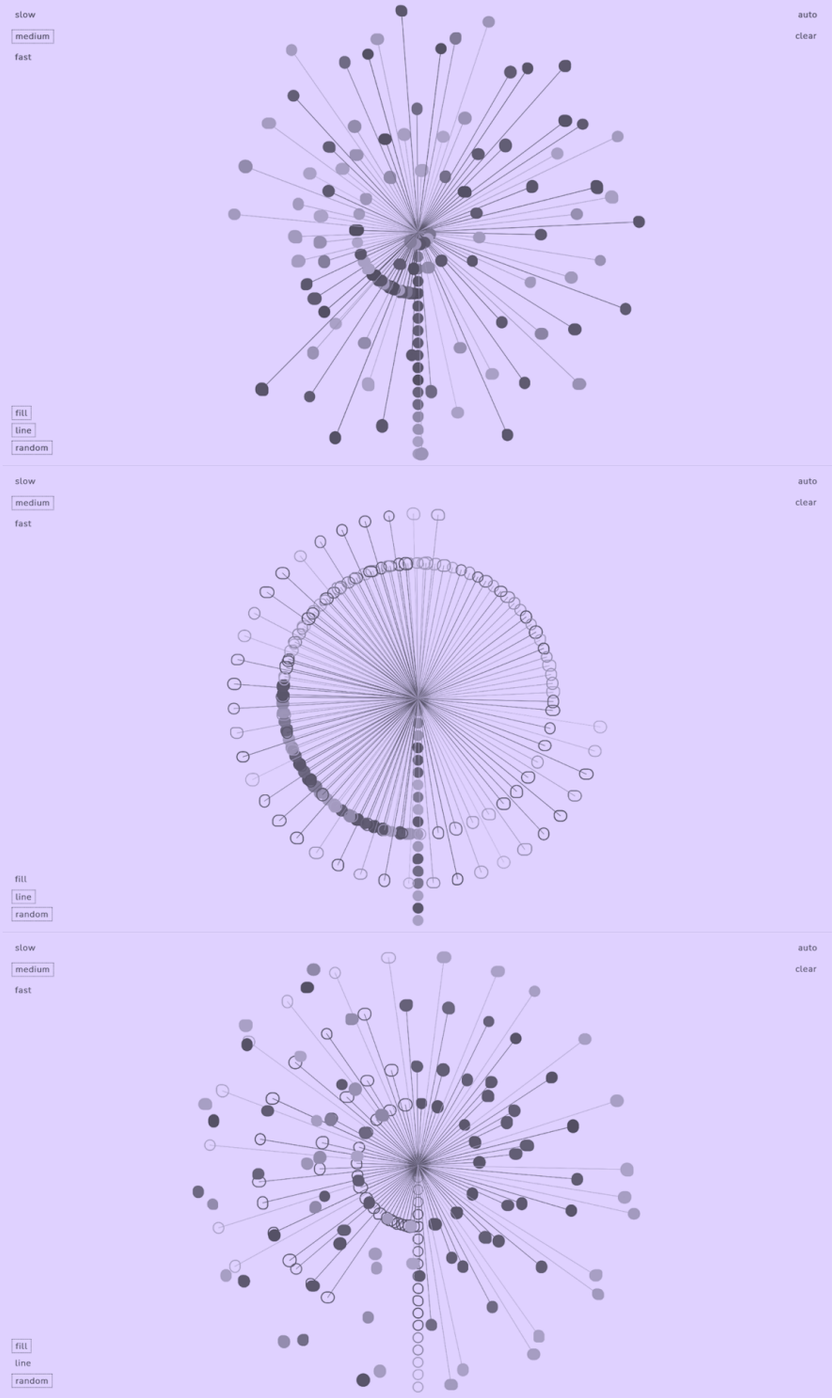




## Rules of Six

*Rules of Six* is an installation commissioned as part of the Design and Elastic Mind exhibition at the Museum of Modern Art in New York curated by Paola Antonelli 2008. In collaboration with material scientist Matthew Scullin, the project explores the notion of self-assembly, where top-down methods for determining form are replaced by bottom up rules of formation; where new material structures are not carved or composed by conventional tools but are "grown" through simple interactions between components or molecules. At the heart of the project is a custom piece of software written *Processing* that simulates formation over time in the same way molecules assemble themselves in the lab. Three-dimensional output from this application was used to produce a large-scale wall relief mounted in the gallery. Like the nanostructures it emulates, Rules of Six is designed to multiply indefinitely without sacrificing stability. It is indifferent to scale; its sprawling construction could represent molecules, rooms, buildings or entire neighborhoods.

Photo Credit: Alex Terzich




# BLOOM ECHO

Bloom Echo is
a generative sound toy.

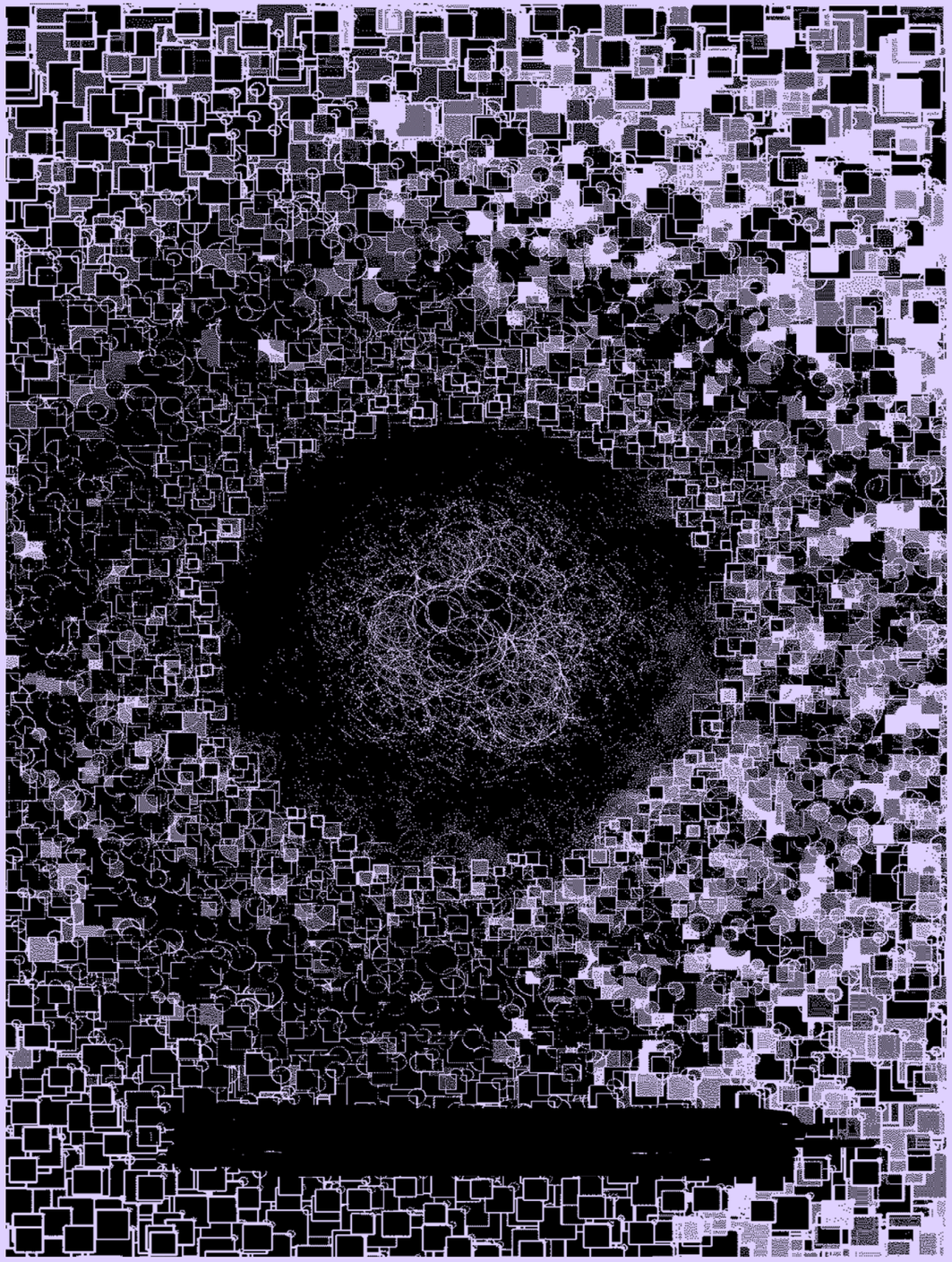
Play with it to chill out
and make blooming,
undulating beats,
or turn on auto + slow
mode to meditate.

Or do both and interact
in auto mode to
create music with
your computer.

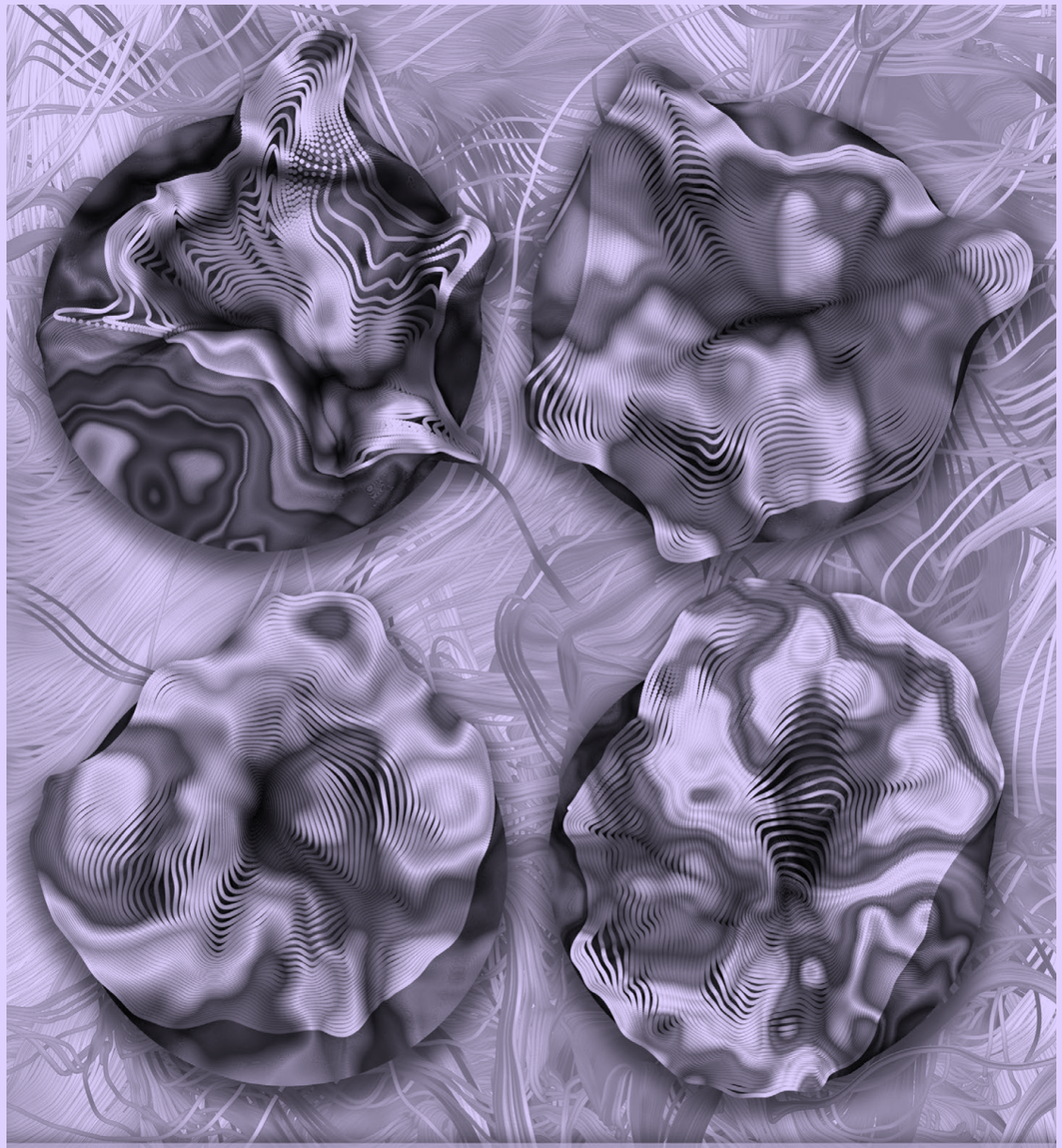
Time spent
with Bloom Echo
is reflected in
a constantly shifting
generative image
on screen.

Made by Aranya
(Ritesh Lala)









# Abstract Perlin Fields

## joey aronson 2021

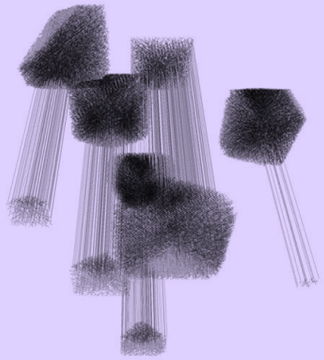
colorful orbs generated using perlin noise rendered with proccesing3
each orb has a render time of about 6 hours

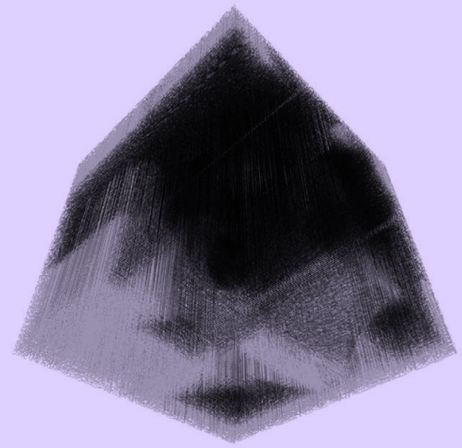
api.artblocks.io/project/0

**Script:**

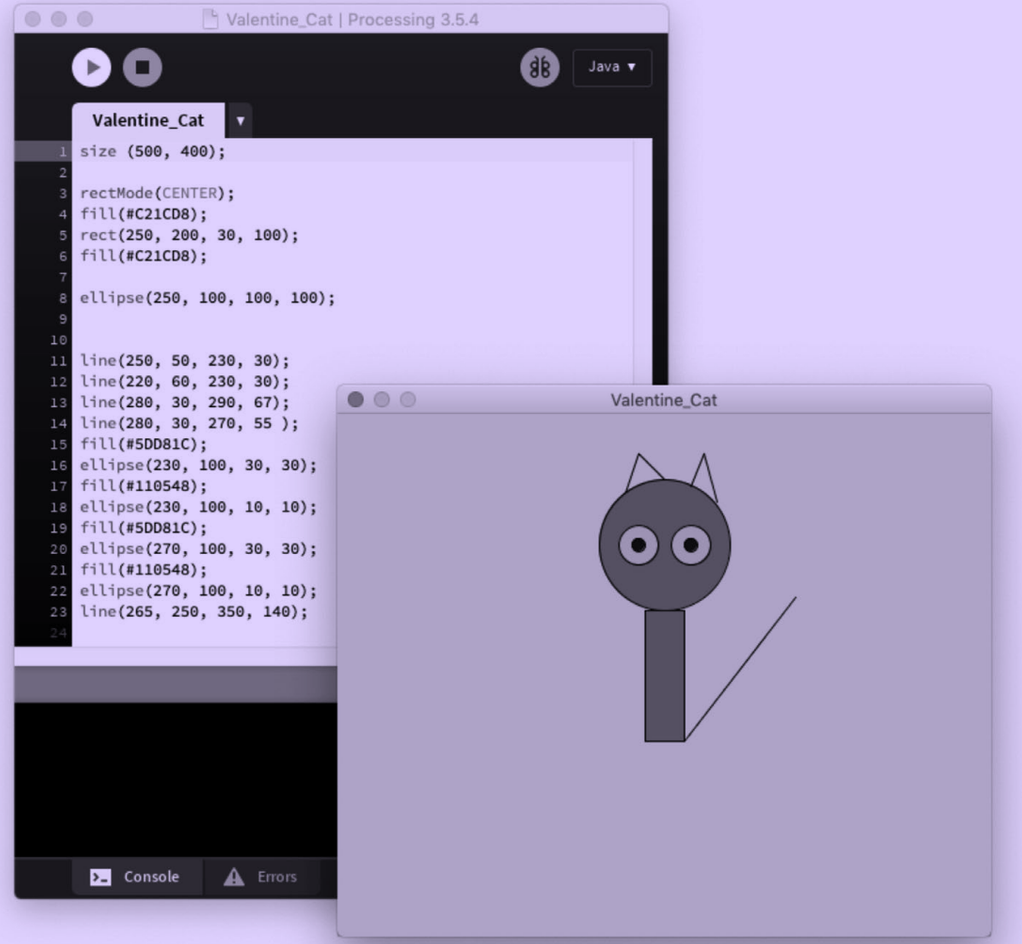
```
let numHashes = tokenData.hashes.length;
let hashPairs = [];
for (let i = 0; i < numHashes; i++) {
     for (let j = 0; j < 32; j++) {
          hashPairs.push(tokenData.hashes[i].slice(2 + (j * 2), 4 + (j * 2)));
     }
}
let decPairs = hashPairs.map(x => {
     return parseInt(x, 16);
});

let seed = parseInt(tokenData.hashes[0].slice(0, 16), 16);
let color;
let backgroundIndex = 0;
let backgroundArray = [255, 225, 200, 175, 150, 125, 100, 75, 50, 25, 0, 25, 50, 75, 100, 125, 150, 175, 200, 225];
let index = 0;
let ht;
let wt = 2;
let speed = 1;
let segments;
let amp = 1;
let direction = 1;
let loops = false;
let startColor = decPairs[29];
let reverse = decPairs[30] < 128;
let slinky = decPairs[31] < 35;
let pipe = decPairs[22] < 32;
let bold = decPairs[23] < 15;
let segmented = decPairs[24] < 30;
let fuzzy = pipe && !slinky;


function setup() {
     let portrait = windowWidth < windowHeight;
     createCanvas(windowWidth > windowHeight * 3 / 2 ? windowHeight * 3 / 2 : windowWidth, windowWidth > windowHeight * 3 / 2 ? windowHeight : wi
     var el = document.getElementsByTagName("canvas")[0];
     el.addEventListener("touchstart", mouseClicked, false);
     colorMode(HSB, 255);
     segments = map(decPairs[26], 0, 255, 12, 20);
     ht = map(decPairs[27], 0, 255, 3, 4);
     spread = decPairs[28] < 3 ? 0.5 : map(decPairs[28], 0, 255, 5, 50);
     strokeWeight(height / 1200);
}

function draw() {
     color = 0;
     background(backgroundArray[backgroundIndex]);
     let div = Math.floor(map(Math.round(decPairs[24]), 0, 230, 3, 20));
     let steps = slinky ? 50 : fuzzy ? 1000 : 200;
     translate((width / 2) - (width / wt / 2), height / 2);
     for (let j = 0; j < segments - 2; j++) {
          for (let i = 0; i <= steps; i++) {
               let t = i / steps;
               let x = curvePoint(width / segments / wt * j, width / segments / wt * (j + 1), width / segments / wt * (j + 2), width / segments /
               let y = curvePoint(map(decPairs[j], 0, 255, -height / ht, height / ht) * amp, map(decPairs[j + 1], 0, 255, -height / ht, height /
               let hue = reverse ? 255 - (((color / spread) + startColor + index) % 255) : (((color / spread) + startColor) + index) % 255;

               if (fuzzy) {
                    noStroke();
                    fill(hue, 255, 255, 20);
                    let fuzzX = x + map(rnd(), 0, 1, 0, height / 10);
                    let fuzzY = y + map(rnd(), 0, 1, 0, height / 10);
                    if (dist(x, y, fuzzX, fuzzY) < height / 11.5) {
                         circle(fuzzX, fuzzY, map(rnd(), 0, 1, height / 160, height / 16));
                    }
               } else {
                    if (slinky && pipe) {
                         if (i == 0 || i == steps - 1) {
                              fill(0);
                         } else {
                              noFill();
                         }
                         stroke(0);
                         circle(x, y, (height / 7))
                    }

                    if (slinky) {
                         if (i == 0 || i == steps - 1) {
                              fill(hue, 255, 255);
                         } else {
                              noFill();
                         }
                         stroke(hue, 255, 255);
                    } else {
                         noStroke();
                         fill(hue, 255, 255);
                    }

                    circle(x, y, bold && !slinky ? height / 5 : height / 13);

                    if (segmented && !slinky && !bold) {
                         if (i % div === 0 || i == 0 || i == steps - 1) {
                              noStroke();
                              fill(decPairs[25]);
                              circle(x, y, height / 12);
                         }
                    }
               }
```

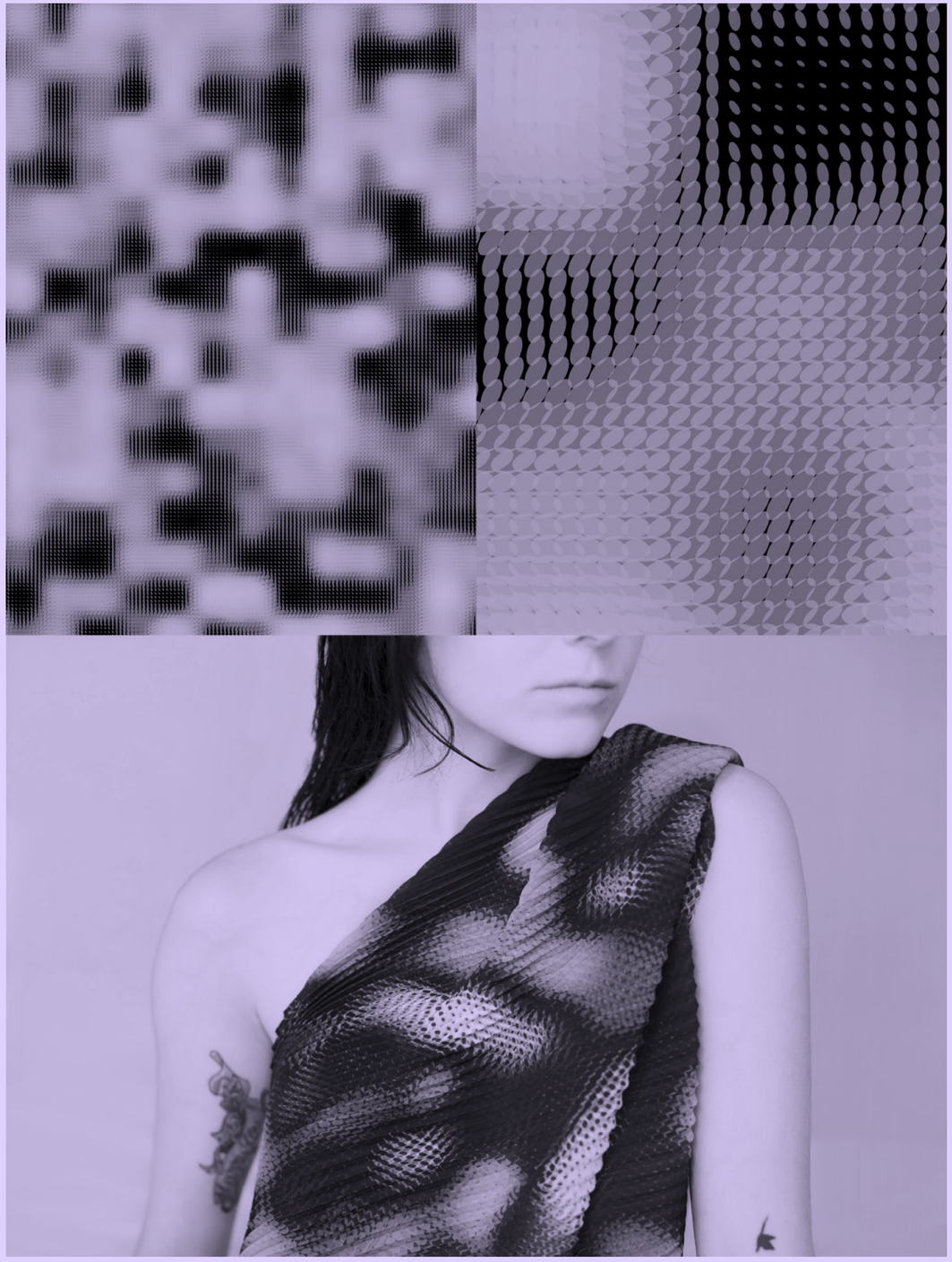


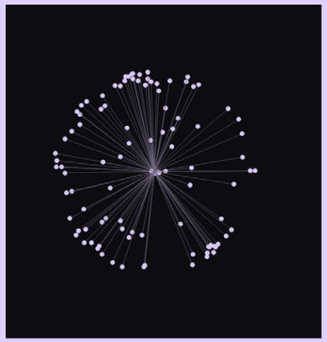


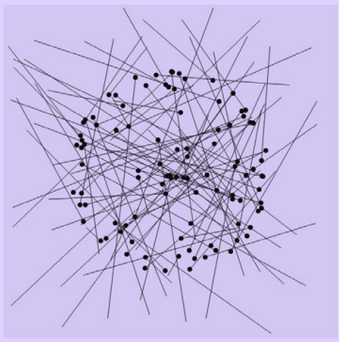


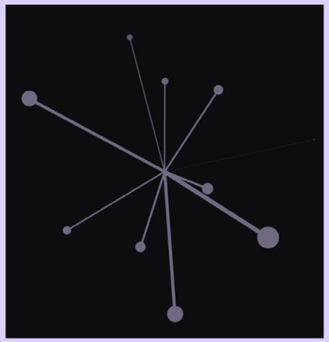
Kim Asendorf, 2010

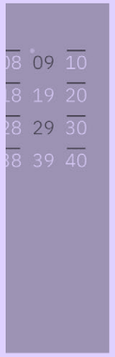

Valentine_Cat | Processing 3.5.4
Java
Valentine_Cat
size (500, 400);
rectMode(CENTER);
fill(#C21CD8);
rect(250, 200, 30, 100);
fill(#C21CD8);
ellipse(250, 100, 100, 100);
line(250, 50, 230, 30);
line(220, 60, 230, 30);
line(280, 30, 290, 67);
line(280, 30, 270, 55 );
fill(#5DD81C);
ellipse(230, 100, 30, 30);
fill(#110548);
ellipse(230, 100, 10, 10);
fill(#5DD81C);
ellipse(270, 100, 30, 30);
fill(#110548);
ellipse(270, 100, 10, 10);
line(265, 250, 350, 140);
Valentine_Cat
Console
Errors

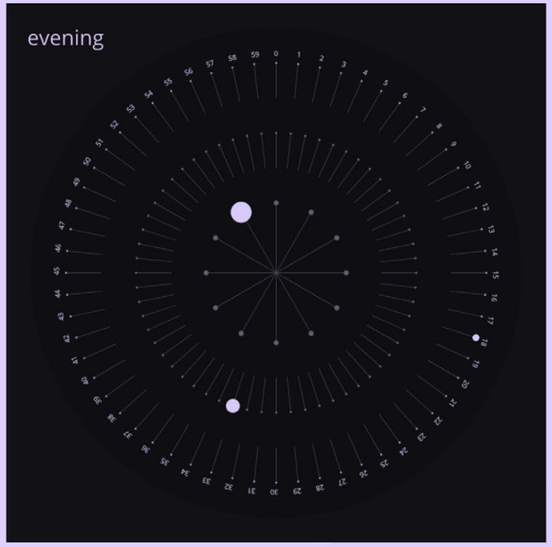


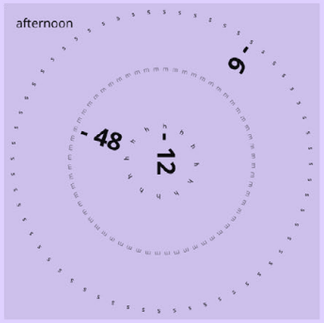




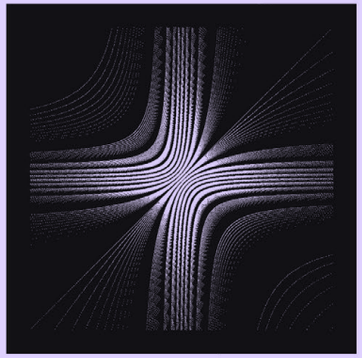


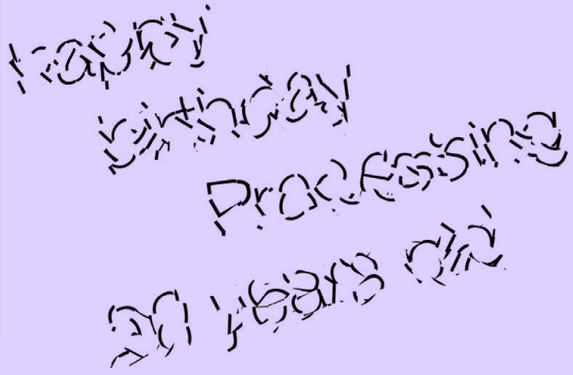





@aymdes

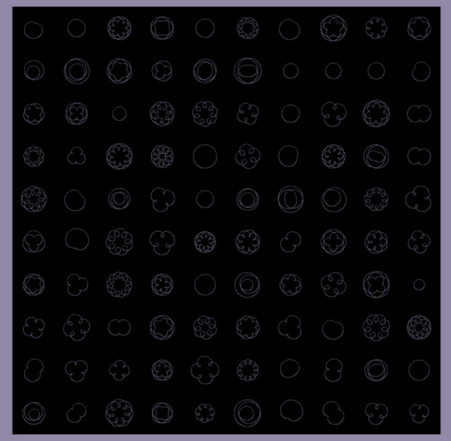






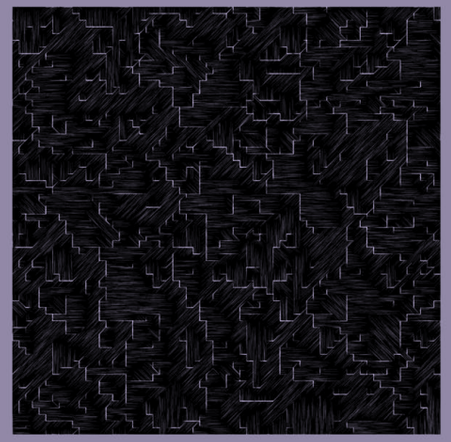

# structure in CHAOS

There is a certain beauty in chaos,
magnificence in the uncontrollable.
**Hafzah Faizal**

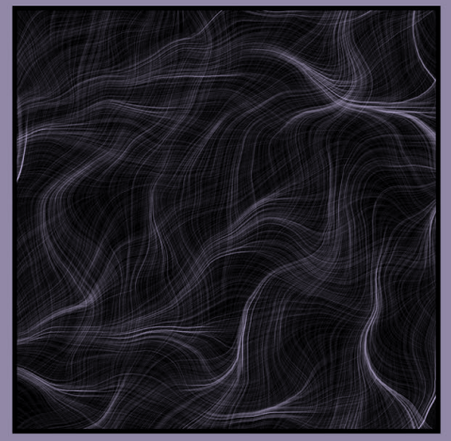


The importance of randomness and chaos is often underestimated. Our universe would not have existed as we know it today without chaos.

Randomness and chaos are critical in making generative art look organic. Such exploration often leads us to discover the structures and patterns that evolve from chaos.



This post is an appreciation of the role pseudo random generators play in creating generative art. Sketches in this page are created using **RANDOM** and **NOISE** functions.



Author - Mitul S Ayyod
Github - github.com/msayyod

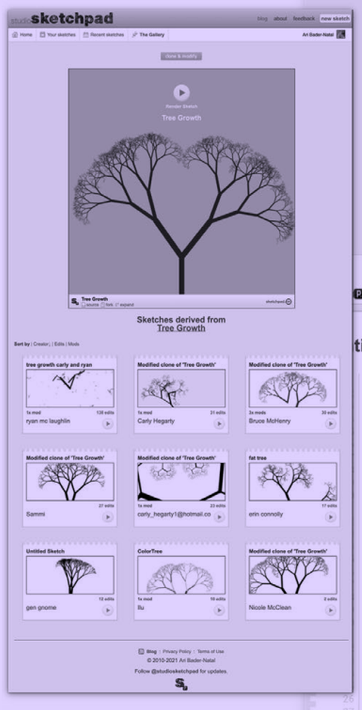
Scan the ARcode to see the augmented content or got to https://b2renger.github.io/processing_catalog

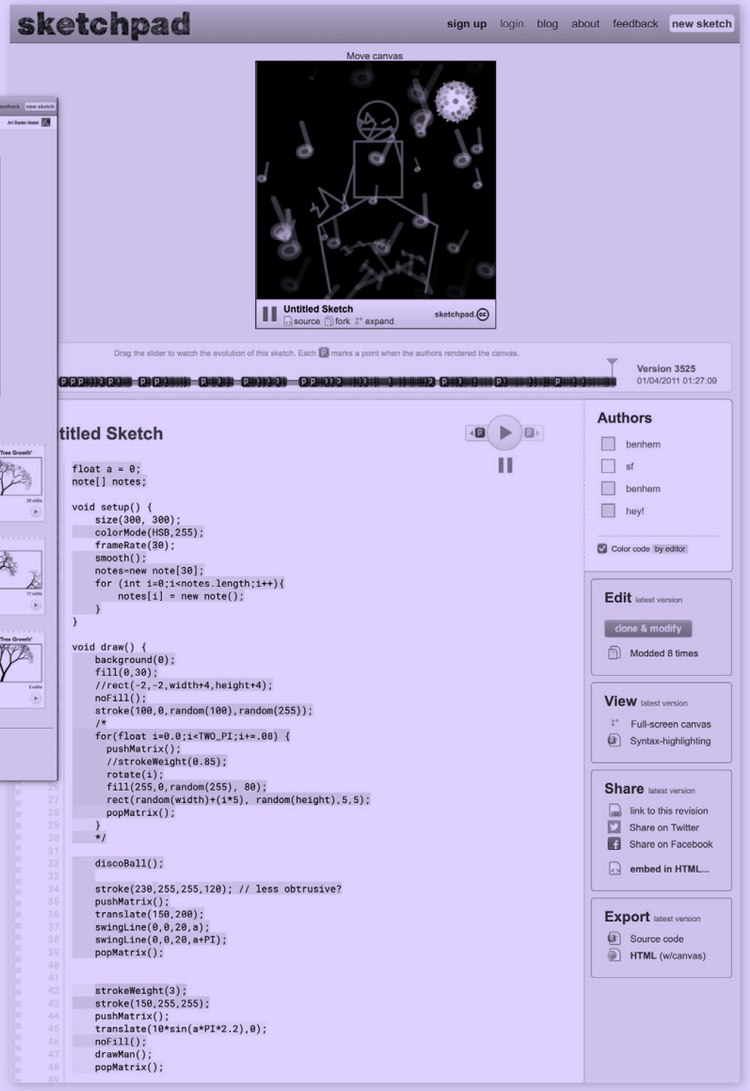


# studio sketchpad

In 2010, sketchpad.cc launched as a site for creative coders to collaboratively create Processing sketches together. Sketchpad repurposed the Etherpad codebase as a real-time collaborative editor for Processing.js (and later for p5.js.) Sketchpad emphasized process over product, giving participants the ability to rewind and replay the creation of any sketch, and to fork and modify the source code from any point in a sketch's history.

Sketchpad was designed with informal learners in mind, but was also used by instructors in over a hundred universities, high schools, and programming workshops. The website was provided as a free (and ad-free) service for over ten years. Congratulations to the Processing community on this 20th anniversary!

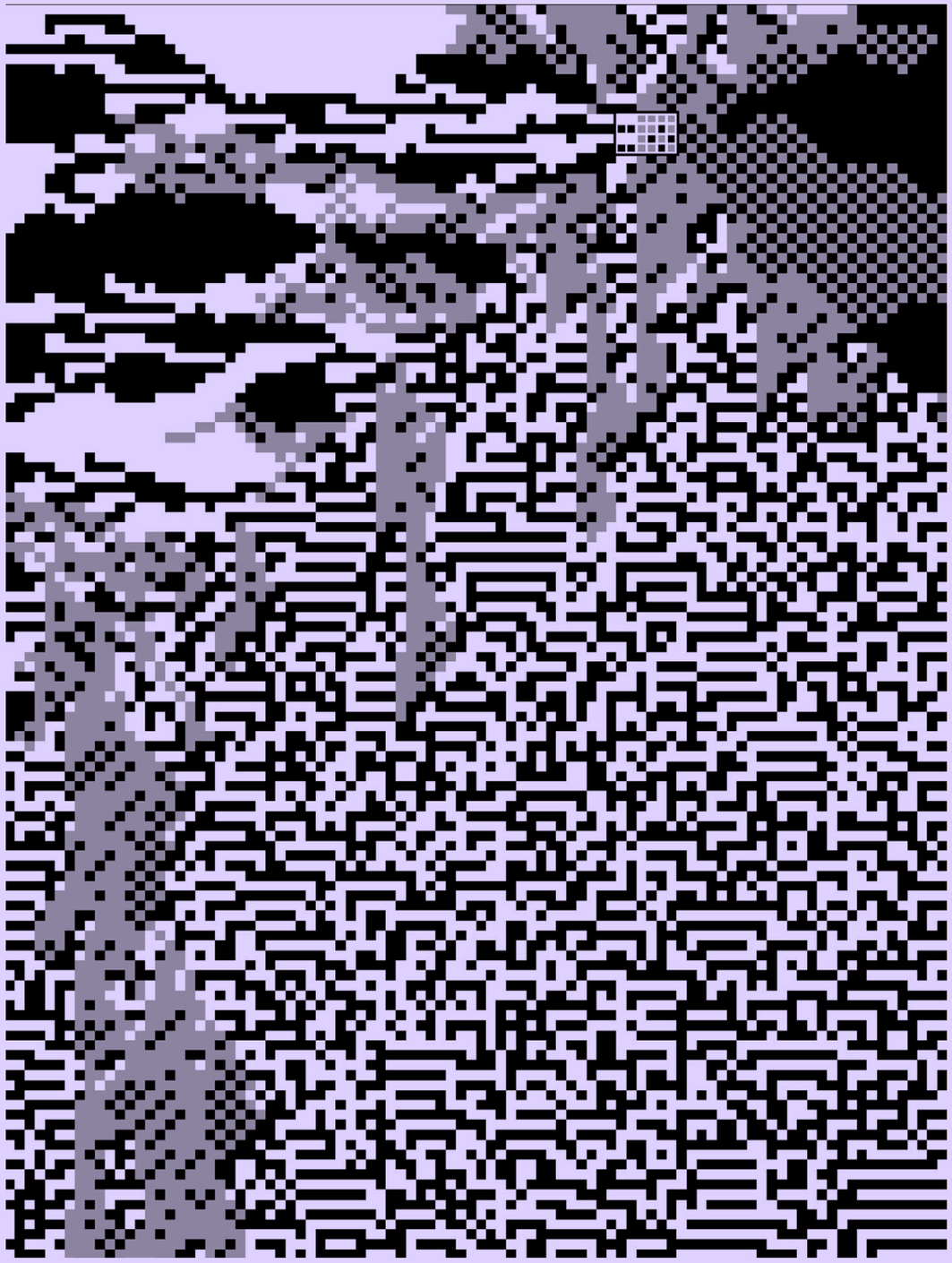



aribadernatal.com
@aribadernatal

sketchpad.cc
@studiosketchpad






Processing Ghent was a series of talks and workshops on Processing, and its many applications in art, design and architecture.

In the final edition we got Peter Beyls over to talk about his experience with creating generative art with early computers in the 1970s, and did a workshop on translating early computer art to the Processing language.

Processing Ghent was organized by Bert Balcaen, Corneel Cannaerts and Jan Vantomme.



Have a Nice Day in Basel

The project is a series of experimental visualizations on local weather. While living in Basel, we observed how the city's climate changes unpredictably.

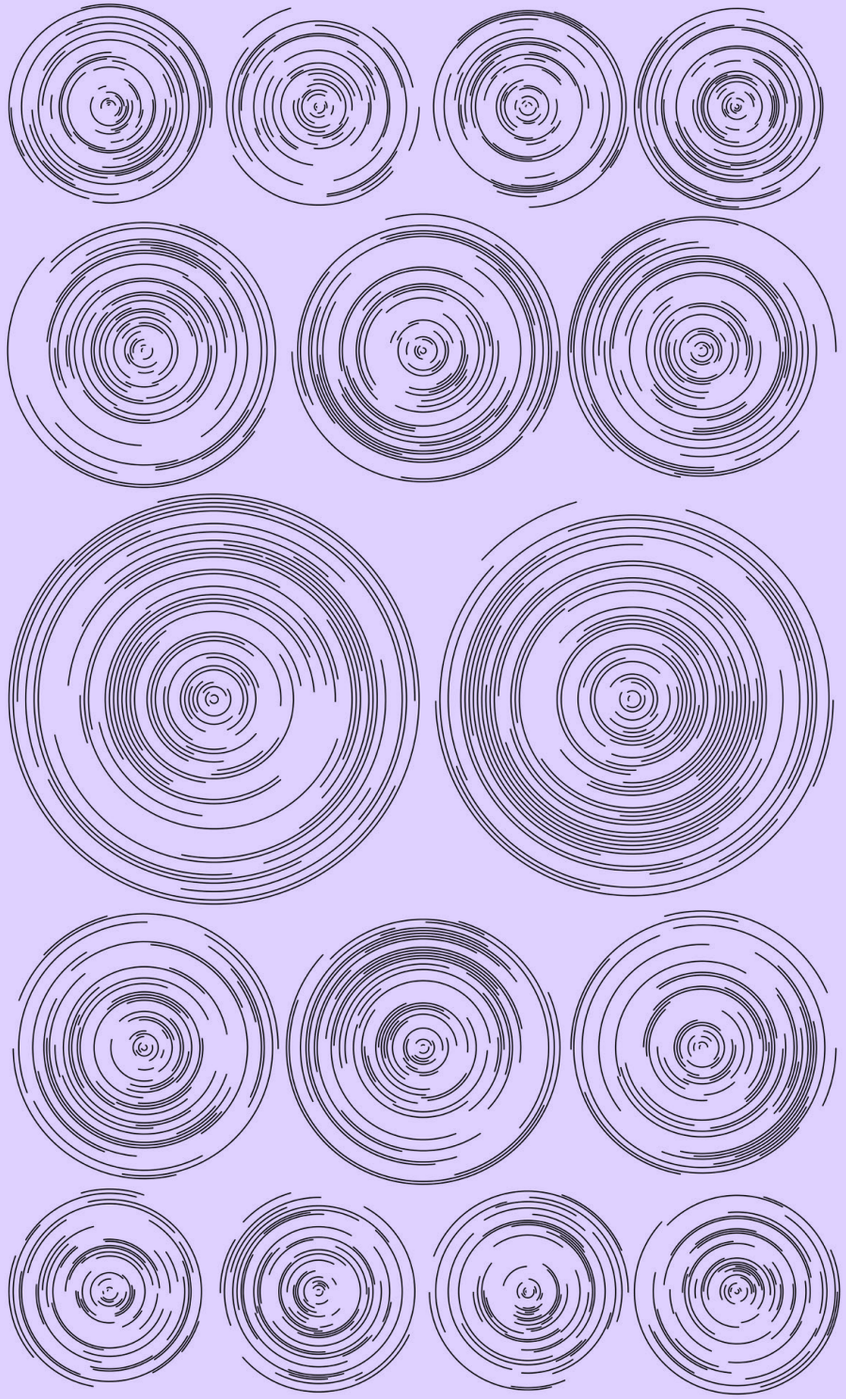
The book is divided into three chapters: a visual summary of the weather during the year 2020 (Ch I - The past), a detailed hour-by-hour analysis of the changes at various locations over a week (Ch II - The Present), and a three weeks visualization of forecasts in the city (Ch III - The Future).

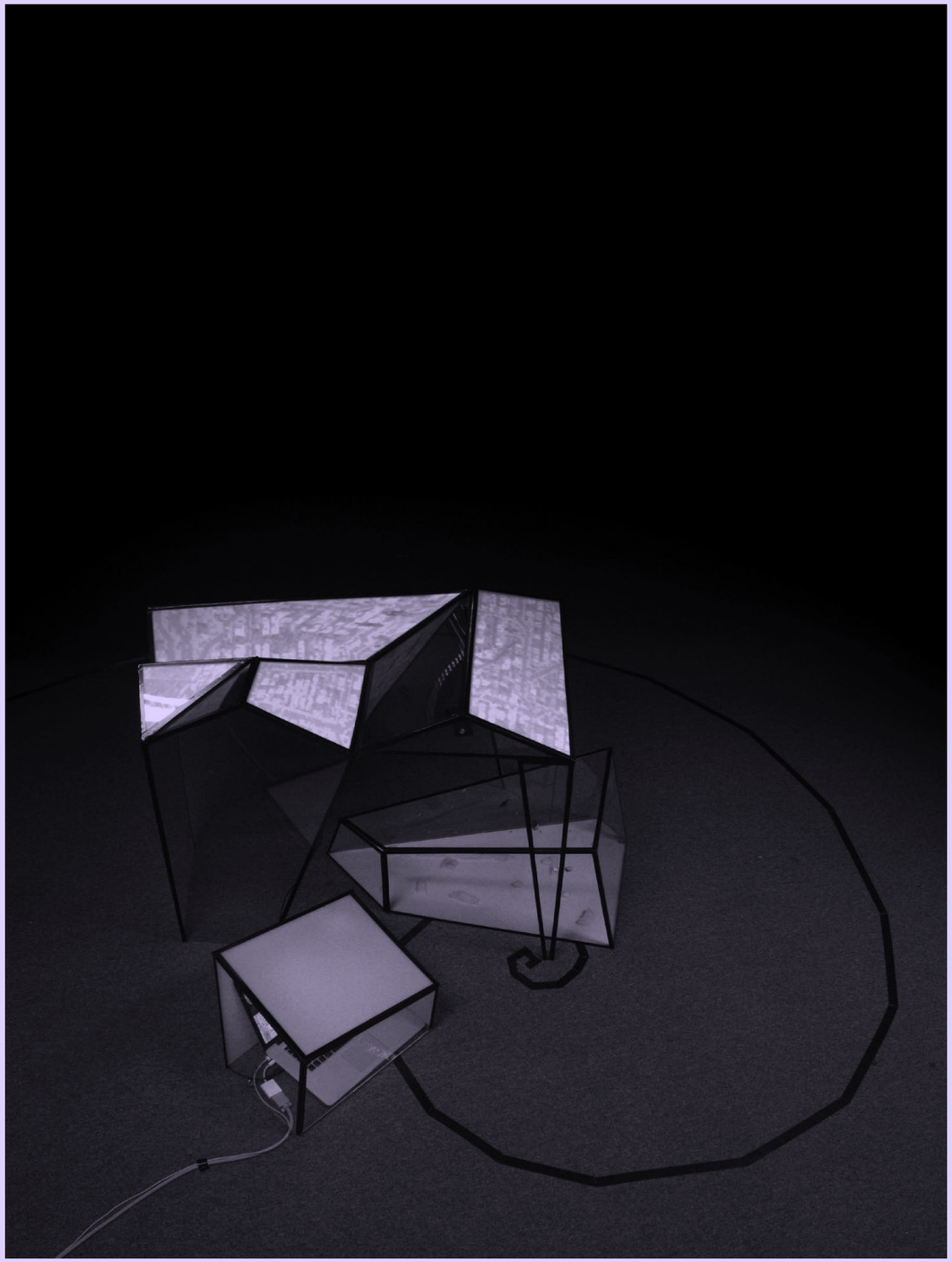
For each chapter we experimented with different language codes, using different databases and data scales. Each section stretch through different time ranges (hours, weeks, year) and space scales (the whole city or specific locations) into consideration, therefore every visualization offers different reading experiences, in which the perception of the data is dilated or concentrated. The user interaction is also various. Going through the pages, the chapters develop through an initial condensed overview of data, then a quick flipping of hourly data and last a schematic calendar-like representation.

Giorgia Bandiera (@knockthedot)and Maria Alessandra Fratta (mafratta.com)

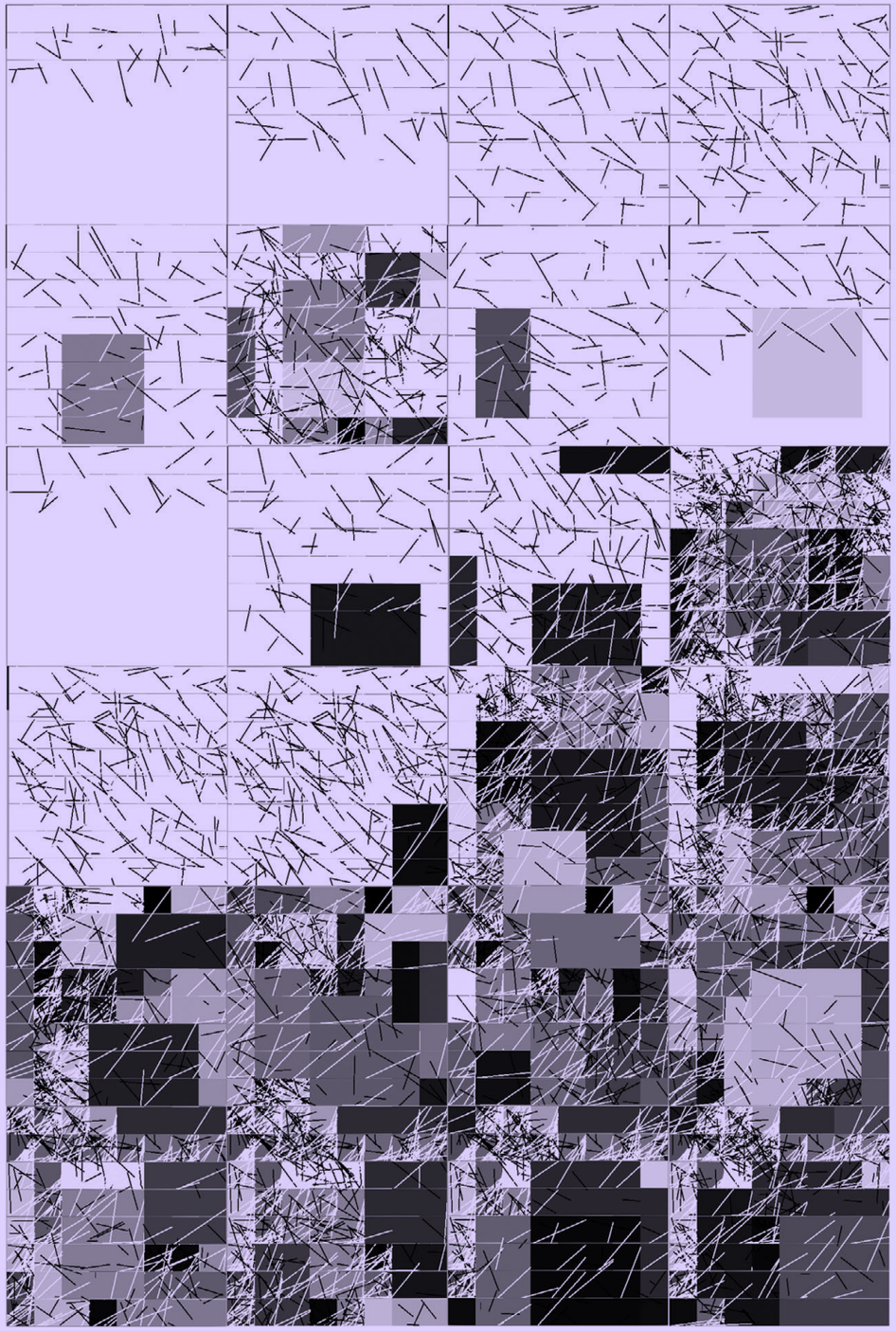

font = loadFont("../includes/demos-data/fonts/RobotoMono-Regular.otf")
} // SOMEHOW NOT WORKING ???????????
function setup() {
createCanvas(windowWidth, windowHeight, WEBGL);
noStroke();
pixelDensity(1)
textFont(font)
}
function draw() {
//background(220);
push()
rotateX(frameCount*0.01)
rotateZ(frameCount*0.01)
rotateY(frameCount*0.01)
stroke(255)
noStroke (5)
textAlign(CENTER, CENTER)
fill (255)
//fill (51, 102, 255)
textSize(250)
//text('day#2', 0, 0)
pop ()
//For (var BEGIN; END; INTERVAL){
//DO SOMETHING }
for (var x = -width/2; x < width; x += width / 10) {
for (var y = -height/2; y < height; y += height / 4) {
//stroke(150);
//strokeWeight(3);
line(x, 0, x, -height/2); //somehow not sure
line(0, y, -width/2, y);
line(x, 0, x, height);
line(0, y, width, y);
fill (0,0,255)
// rect (x, y, 10,50) --> rect is working
push()
rotate (frameCount*0.0007%width)
ambientLight(200);
ambientMaterial (0, 50, 255);
stroke(255);
strokeWeight(1);
translate(x,y)
//cone(20, 160);











To Archivists of the Future

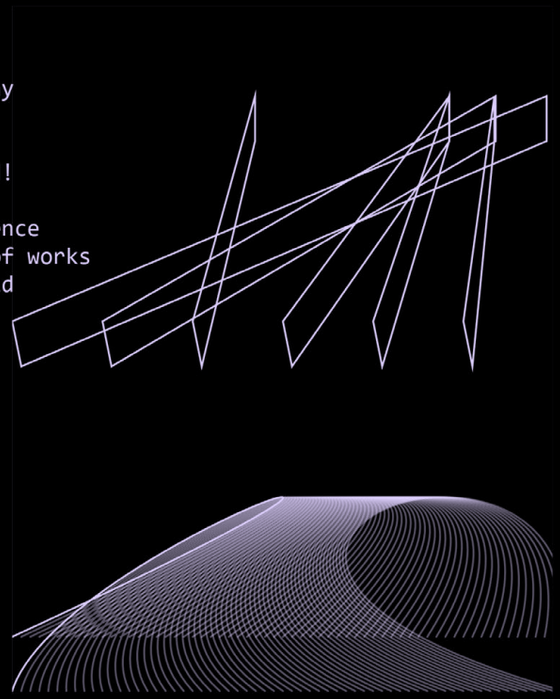
Hello, World!

The image on this page
[the first, for me
of many marks]
feels like
Broad arts building
burnt coffee elevator
sunshine balcony fourth floor
university of california los angeles

linen rag paper and
the twenty-first century
the second decade
+ trying to learn the new tool
from the new tool master

Platforms are *most* powerful
using the tool has
shaped / formed / fashioned
my own work and thought
on media, culture, philosophy
aesthetics & politics

for the tool shapes the hand!
and
the tool has taken up residence
sticky mortar in the walls of works
that travel in the vast world
digital vectors become
material vectors
of which I am but one
or many



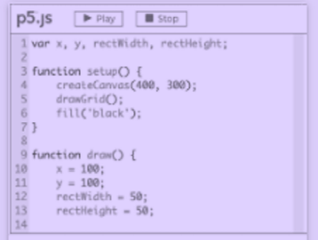
M. James Becker
b. St. Louis, MO, USA, 1994
d. ???
October 1, 2021



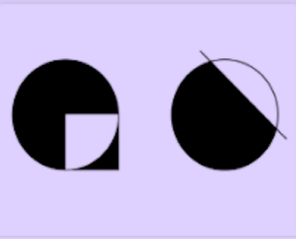
georgesb.com

# Project Notebook

P5.JS NOTEBOOK




CODING SANDBOX

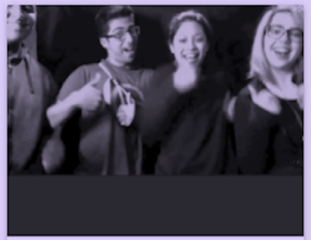


DESIGN + CODE

SONNET

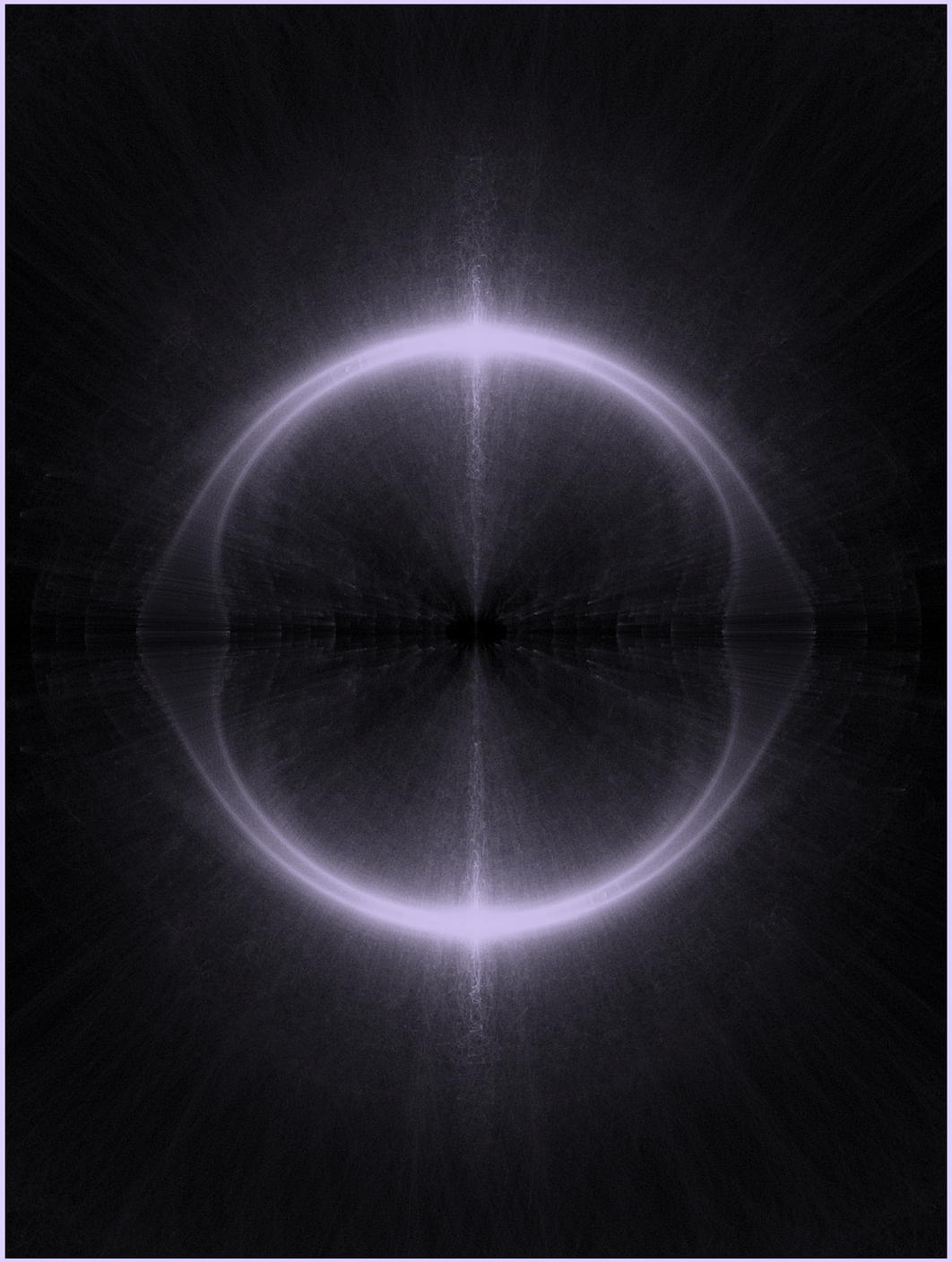
SHAKESPEARE


IMOVIE NARRATION

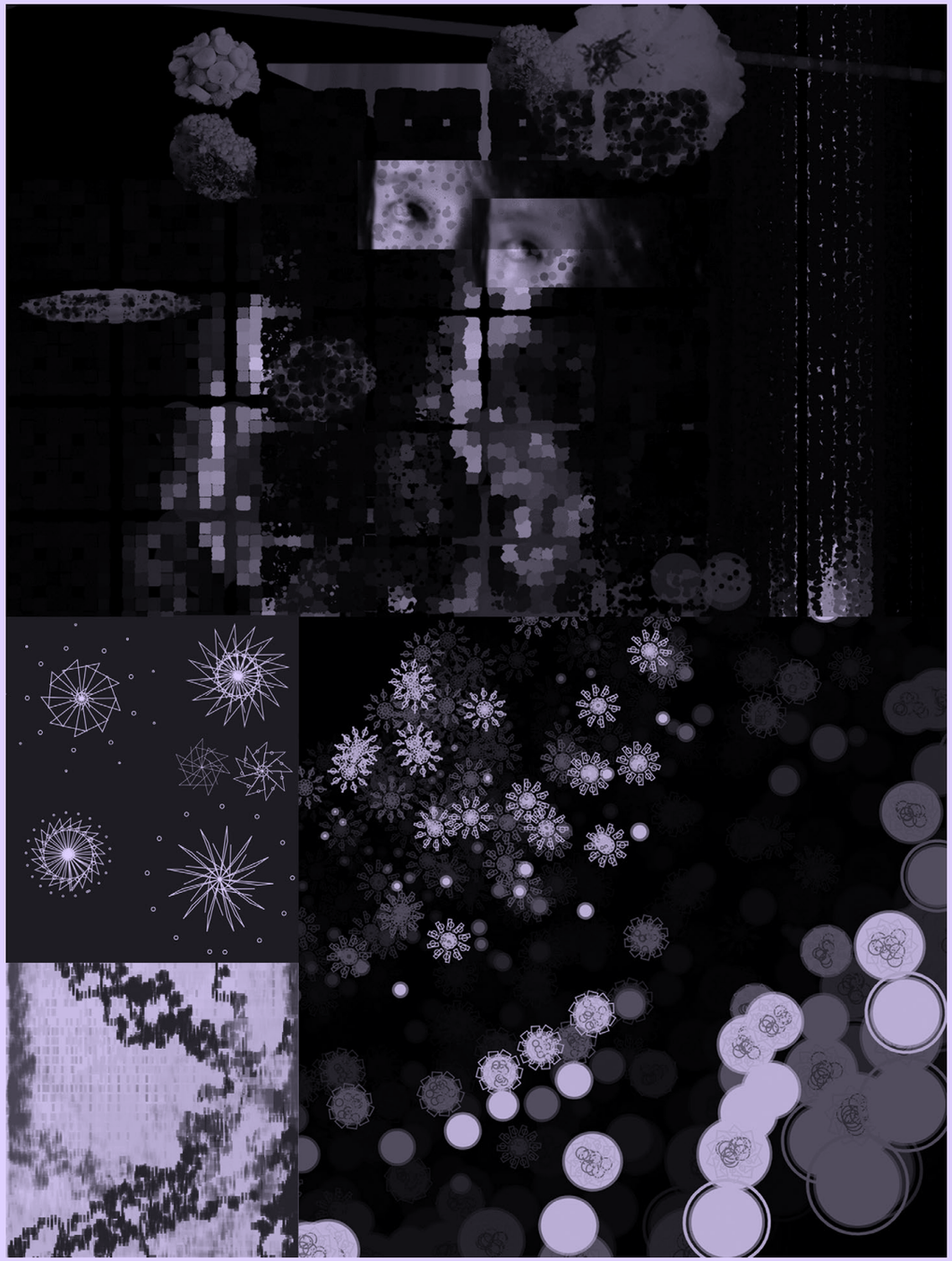


STEM CONTEST

view all

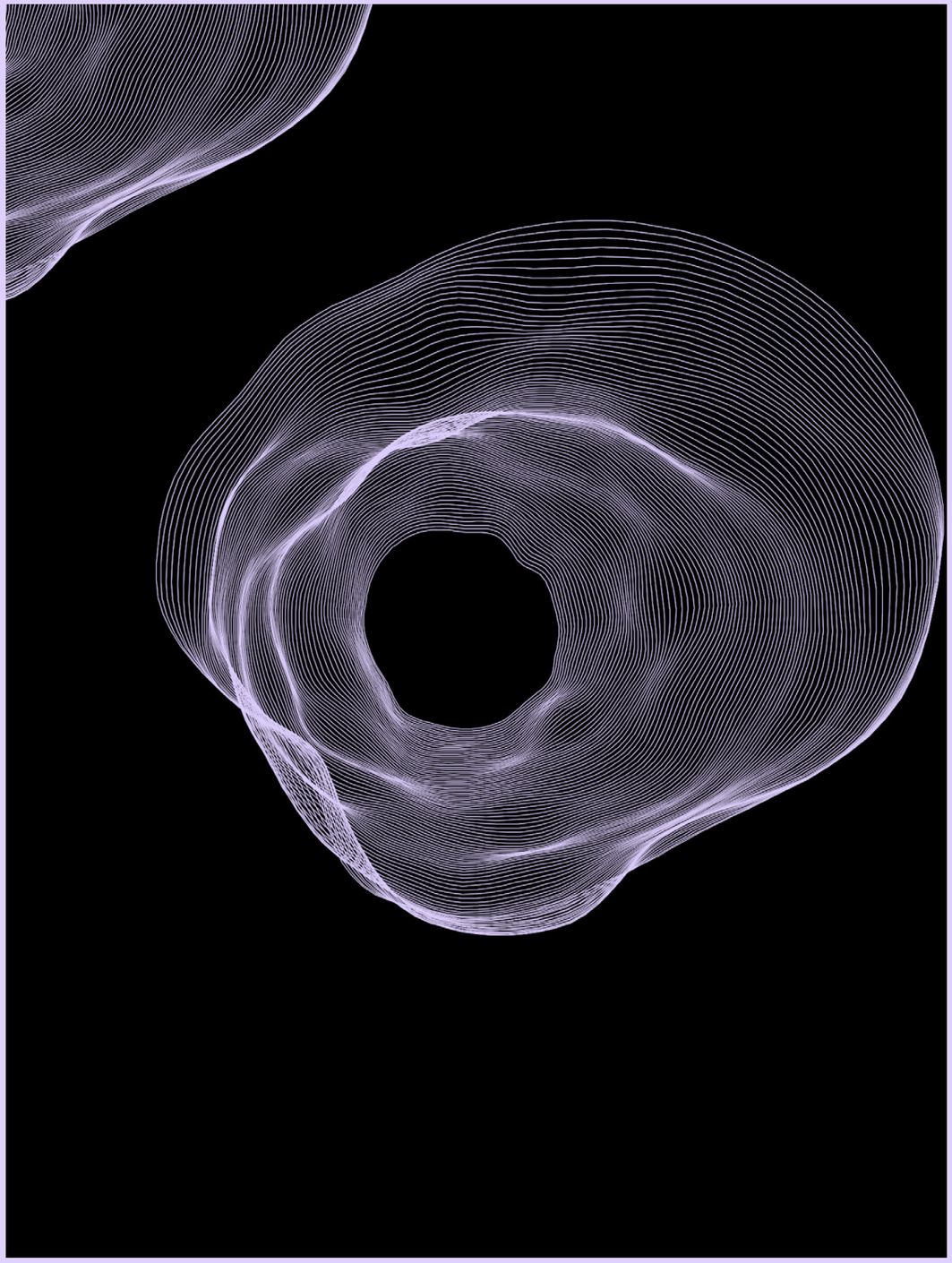


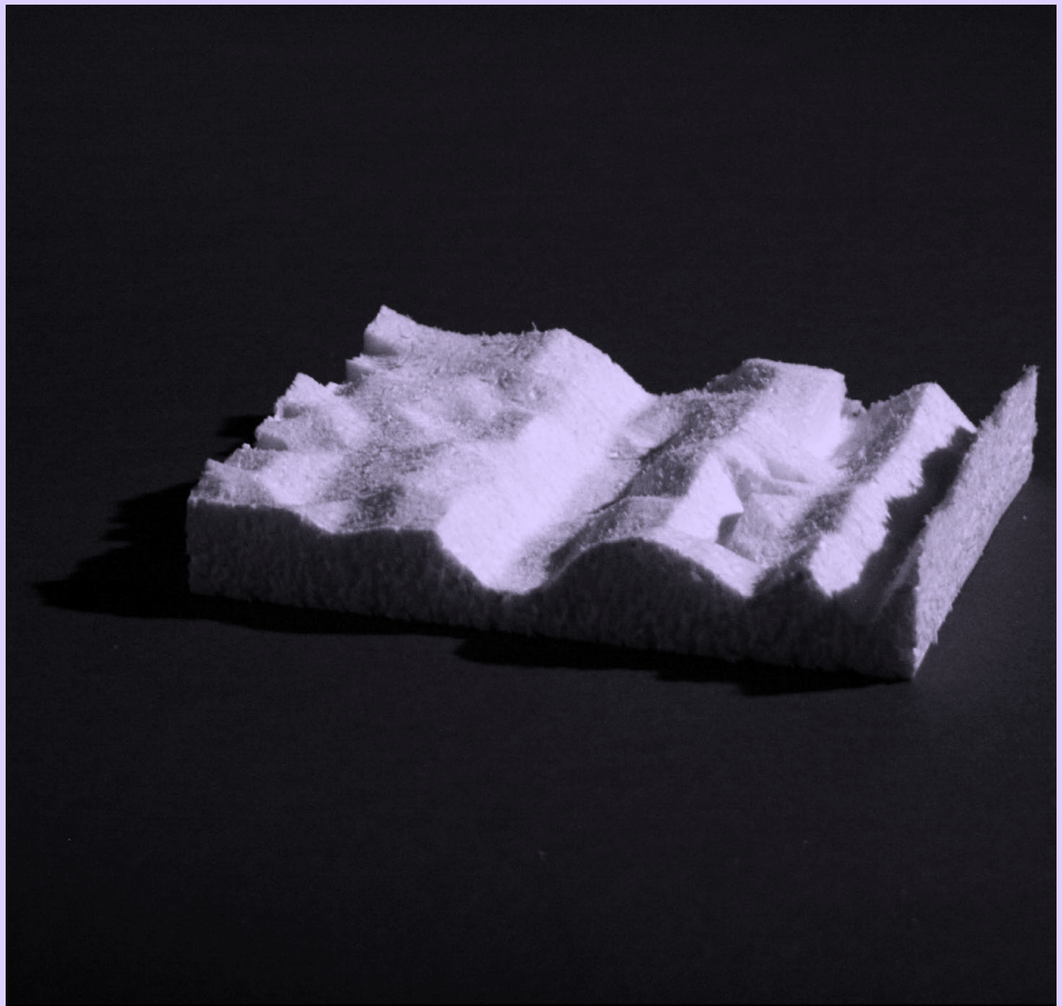




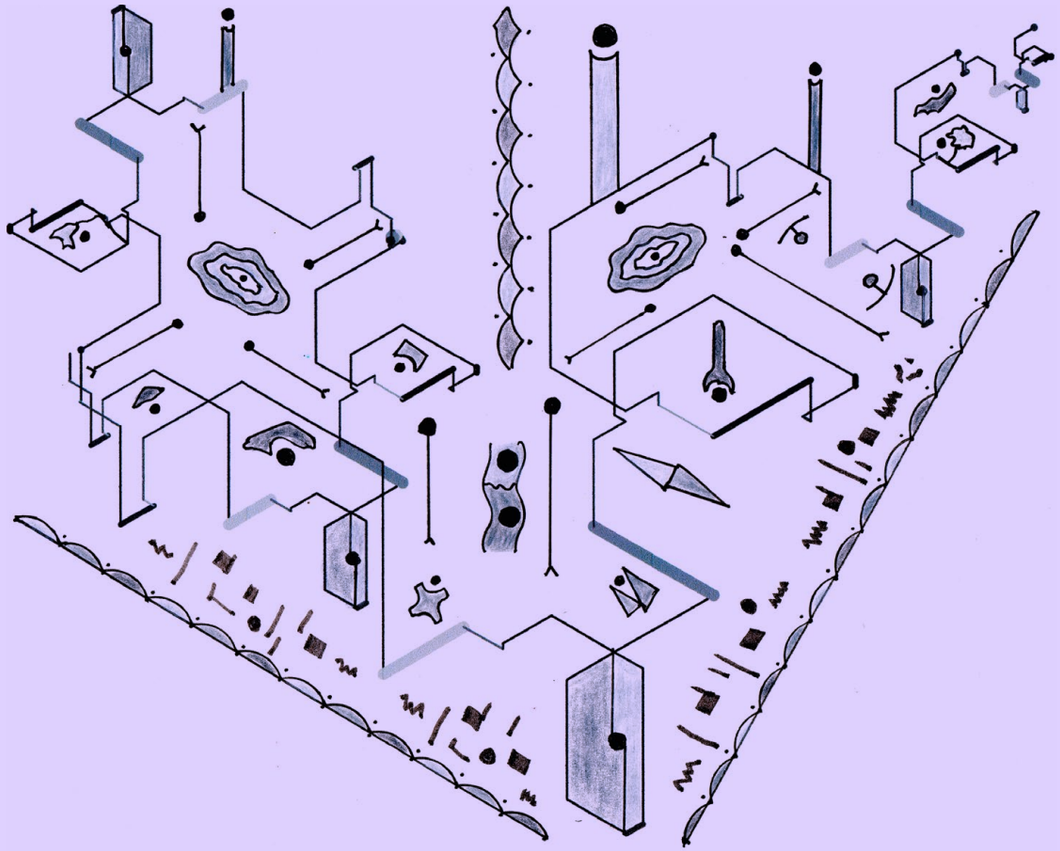

Inspired by Braitenberg and Brockmann
"Thank you
Processing for
allowing patterns to
express themselves."





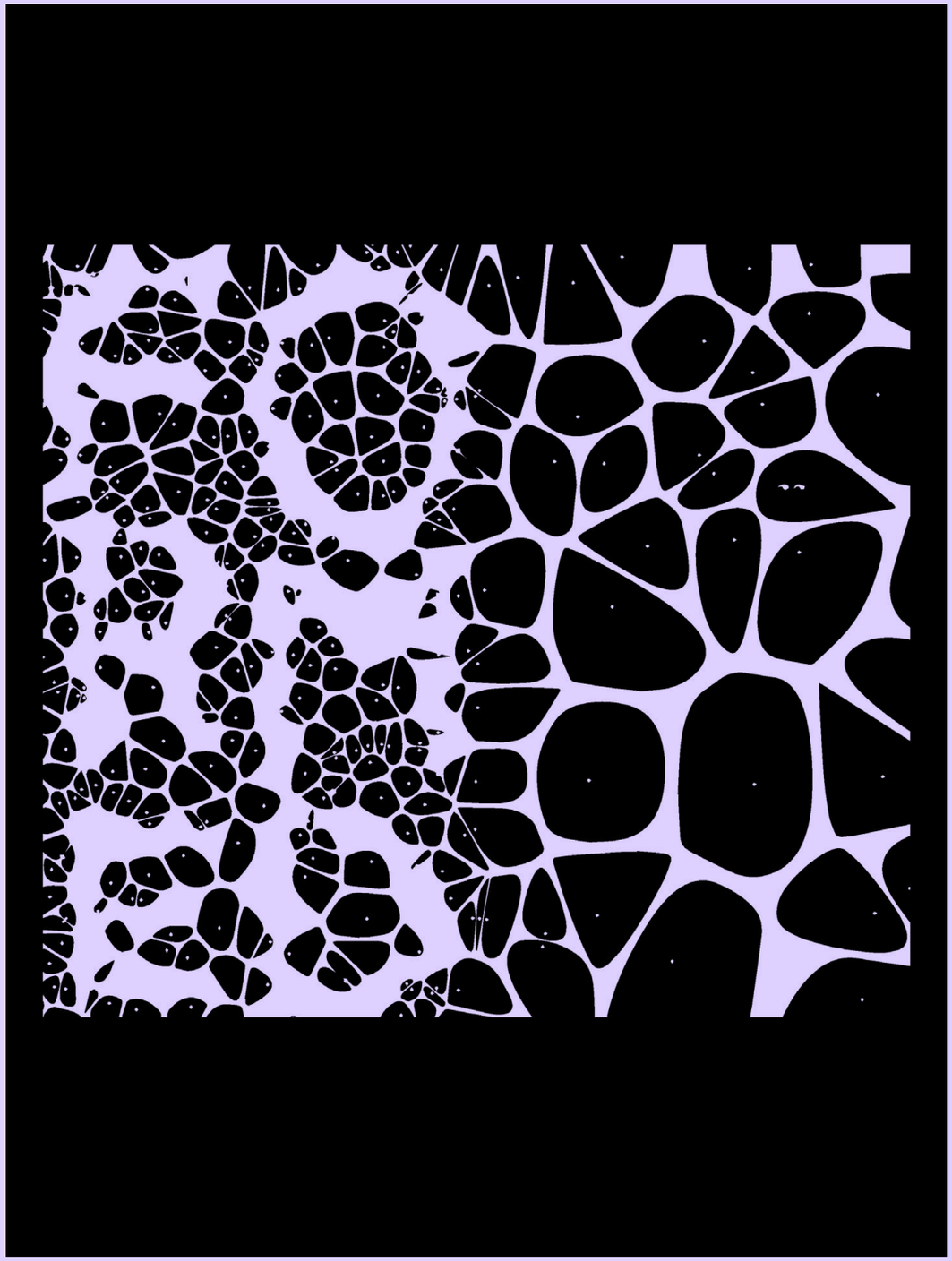


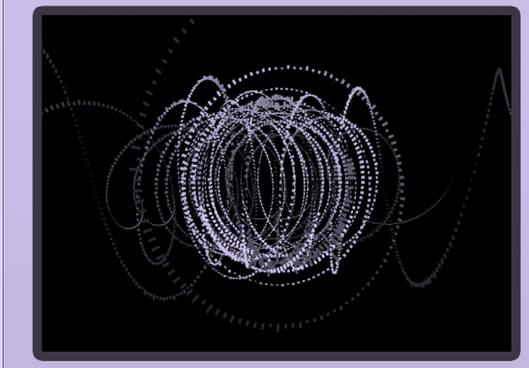


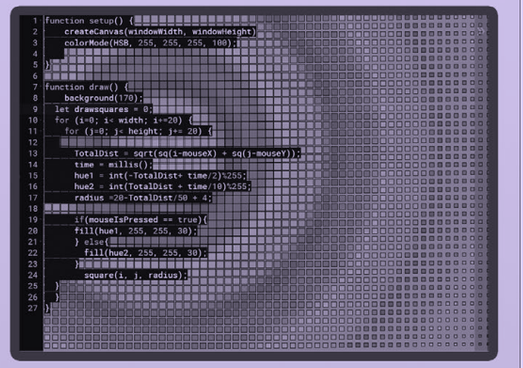



vinyl+
Extended Timecode Vinyl

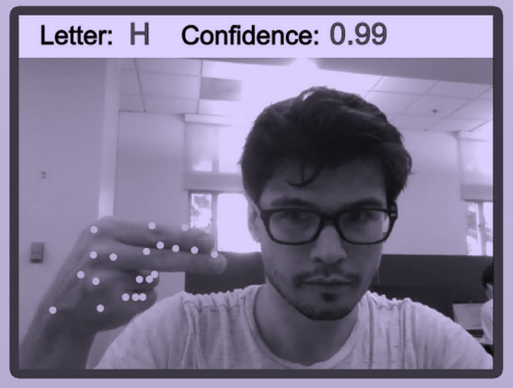




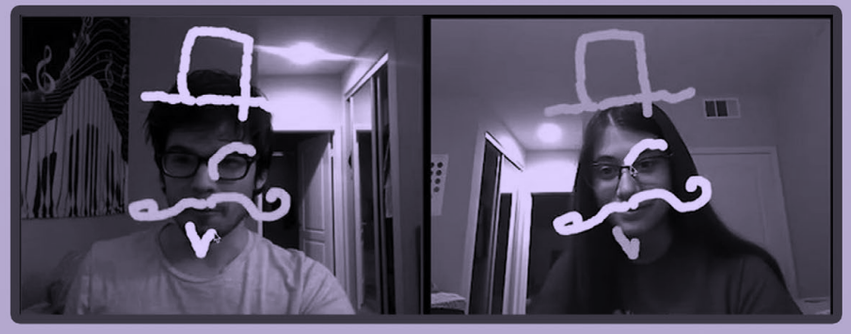




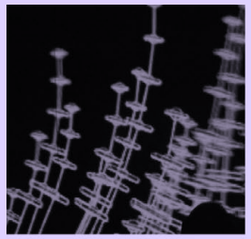
Processing opened my eyes to the world of creative code

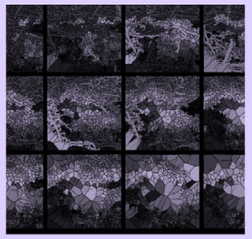


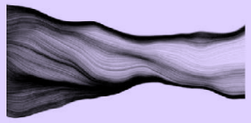

@jay.borgwardt

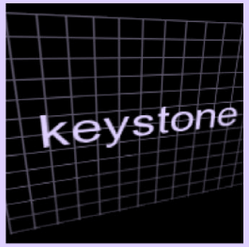


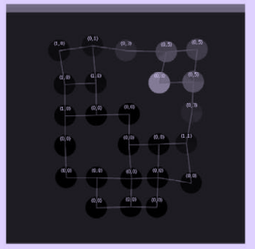


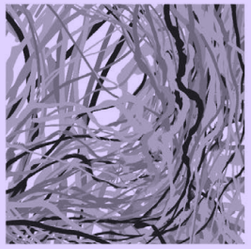


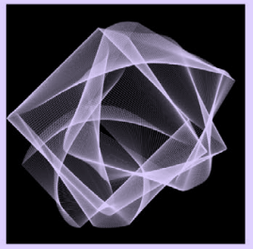


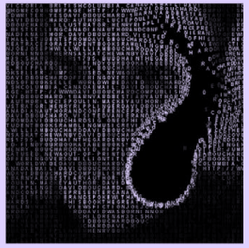


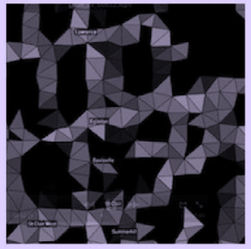


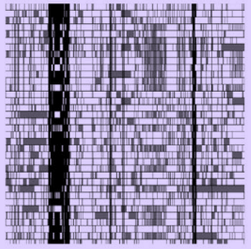







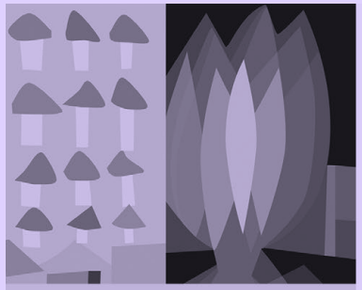


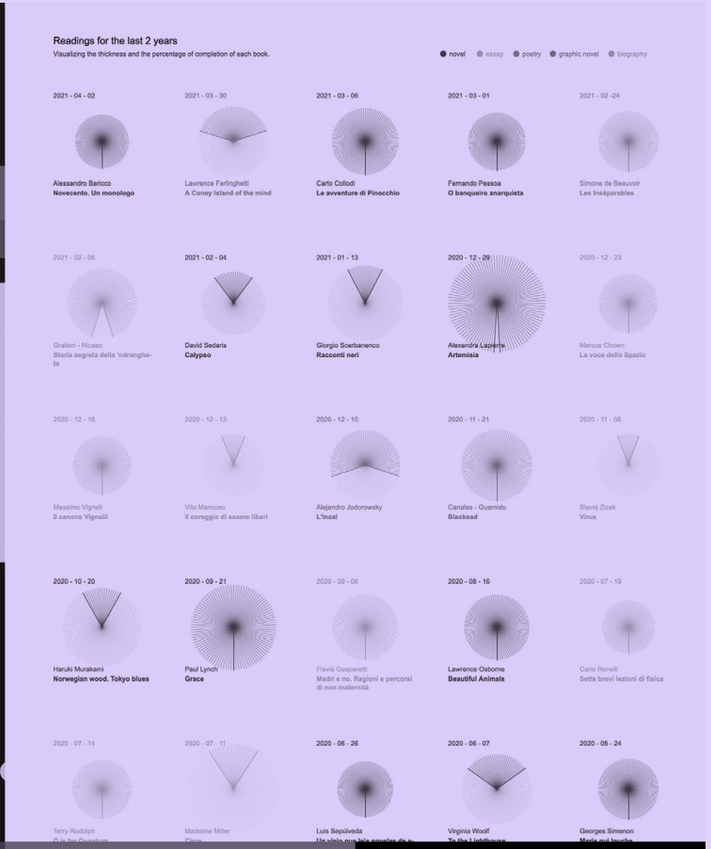


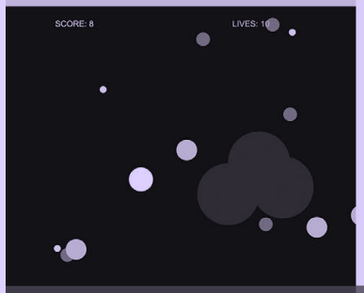


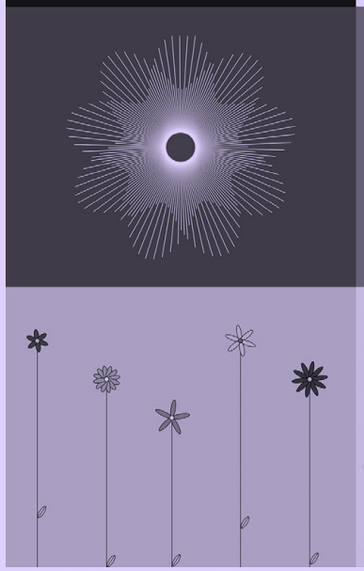
MAKING WITH PROCESSING SINCE 2003

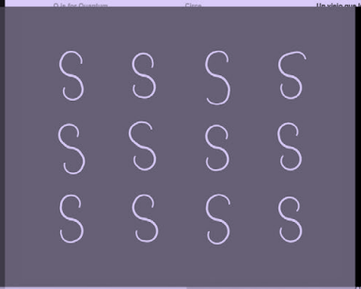
dead★pixel

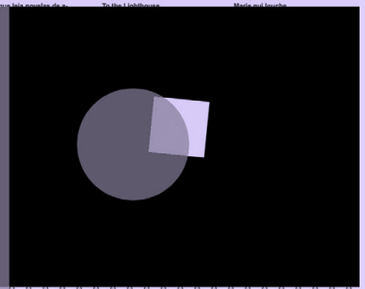



Readings for the last 2 years
Visualizing the thickness and the percentage of completion of each book.
novel
essay
poetry
graphic novel
biography
2021 - 04 - 02
Alessandro Baricco
Novecento. Un monologo
2021 - 03 - 30
Lawrence Ferlinghetti
A Coney Island of the mind
2021 - 03 - 06
Carlo Collodi
Le avventure di Pinocchio
2021 - 03 - 01
Fernando Pessoa
O banqueiro anarquista
2021 - 02 -24
Simone de Beauvoir
Les Inséparables
2021 - 02 - 06
Gratteri - Nicaso
Storia segreta della 'ndrangheta
2021 - 02 - 04
David Sedaris
Calypso
2021 - 01 - 13
Giorgio Scerbanenco
Racconti neri
2020 - 12 - 29
Alexandra Lapierre
Artemisia
2020 - 12 - 23
Marcus Chown
La voce dello Spazio
2020 - 12 - 18
Massimo Vignelli
Il canone Vignelli
2020 - 12 - 13
Vito Mancuso
Il coraggio di essere liberi
2020 - 12 - 10
Alejandro Jodorowsky
L'Incal
2020 - 11 - 21
Canales - Guarnido
Blacksad
2020 - 11 - 08
Slavoj Zizek
Virus
2020 - 10 - 20
Haruki Murakami
Norwegian wood. Tokyo blues
2020 - 09 - 21
Paul Lynch
Grace
2020 - 09 - 08
Flavia Gasperetti
Madri e no. Ragioni e percorsi di non maternità
2020 - 08 - 16
Lawrence Osborne
Beautiful Animals
2020 - 07 - 19
Carlo Rovelli
Sette brevi lezioni di fisica
2020 - 07 - 14
Terry Rudolph
2020 - 07 - 11
Madeline Miller
2020 - 06 - 26
Luis Sepúlveda
2020 - 06 - 07
Virginia Woolf
2020 - 05 - 24
Georges Simenon

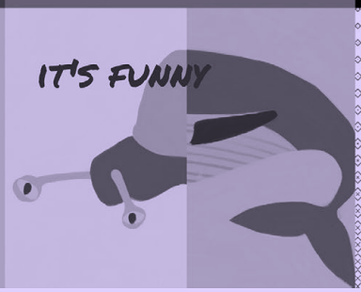

SCORE: 8
LIVES: 1

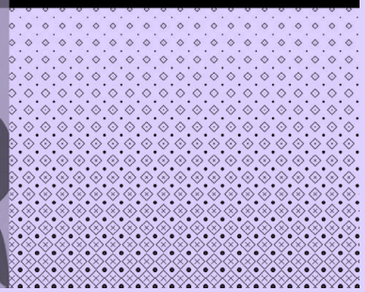




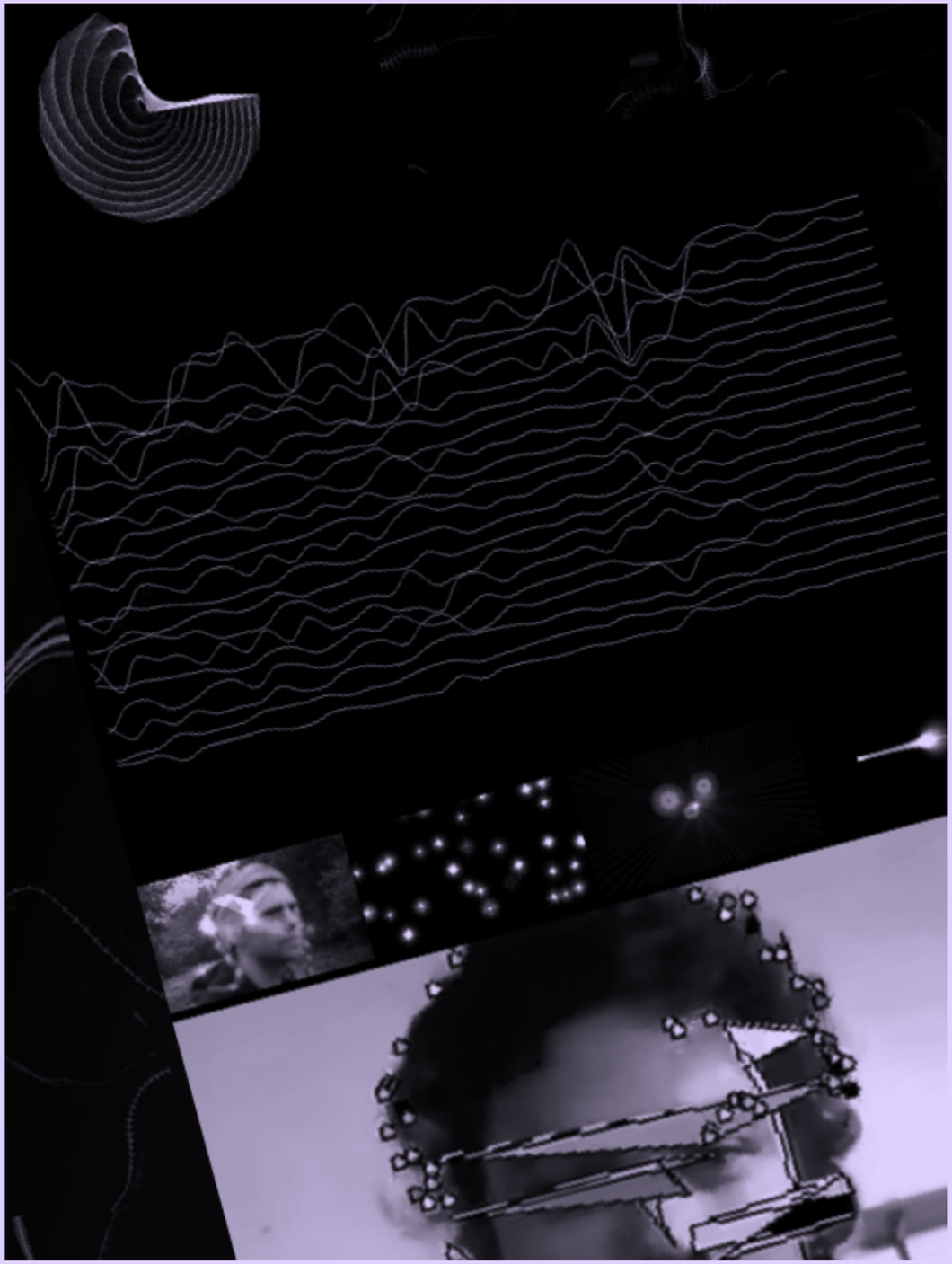


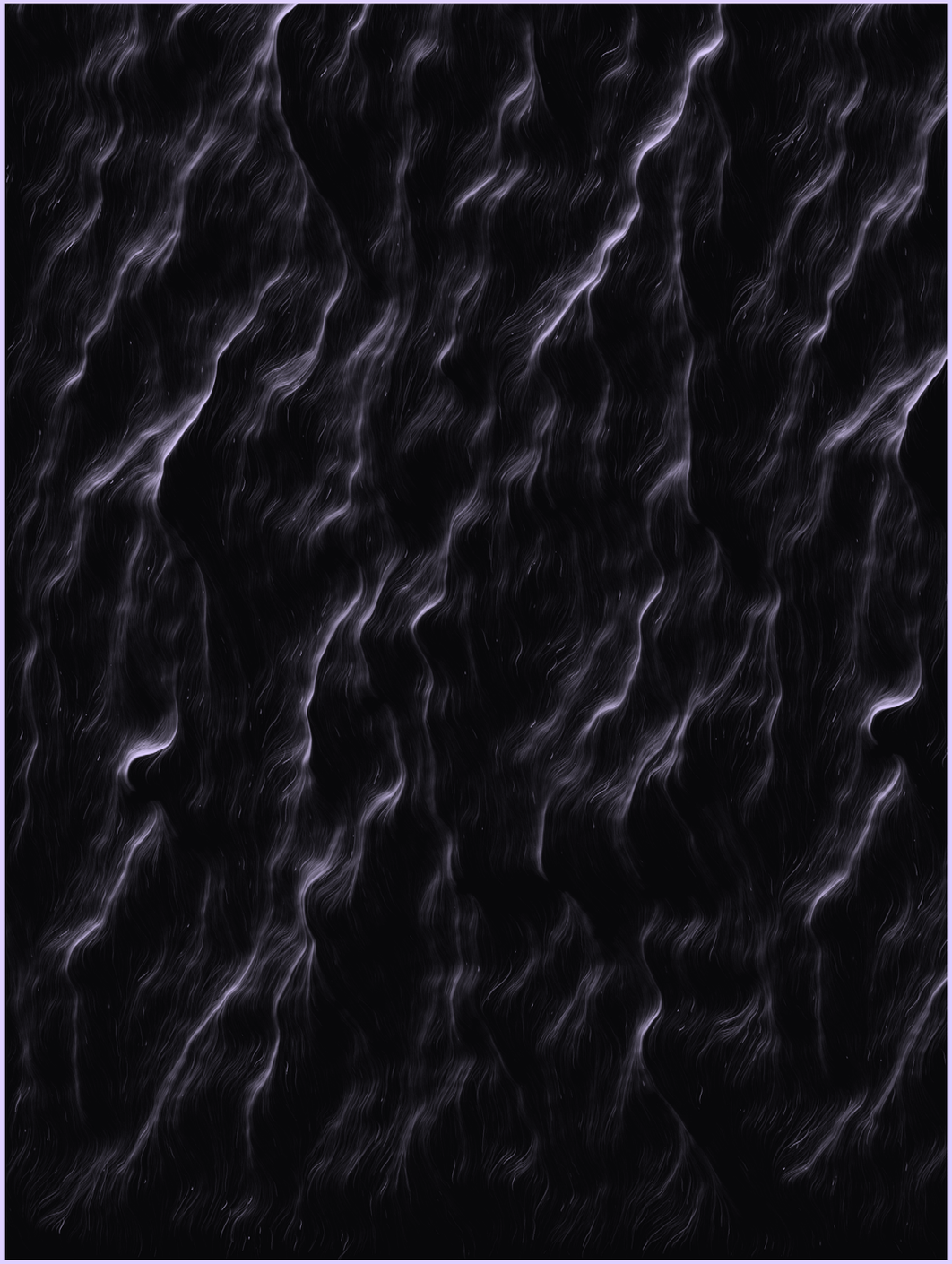



IT'S FUNNY

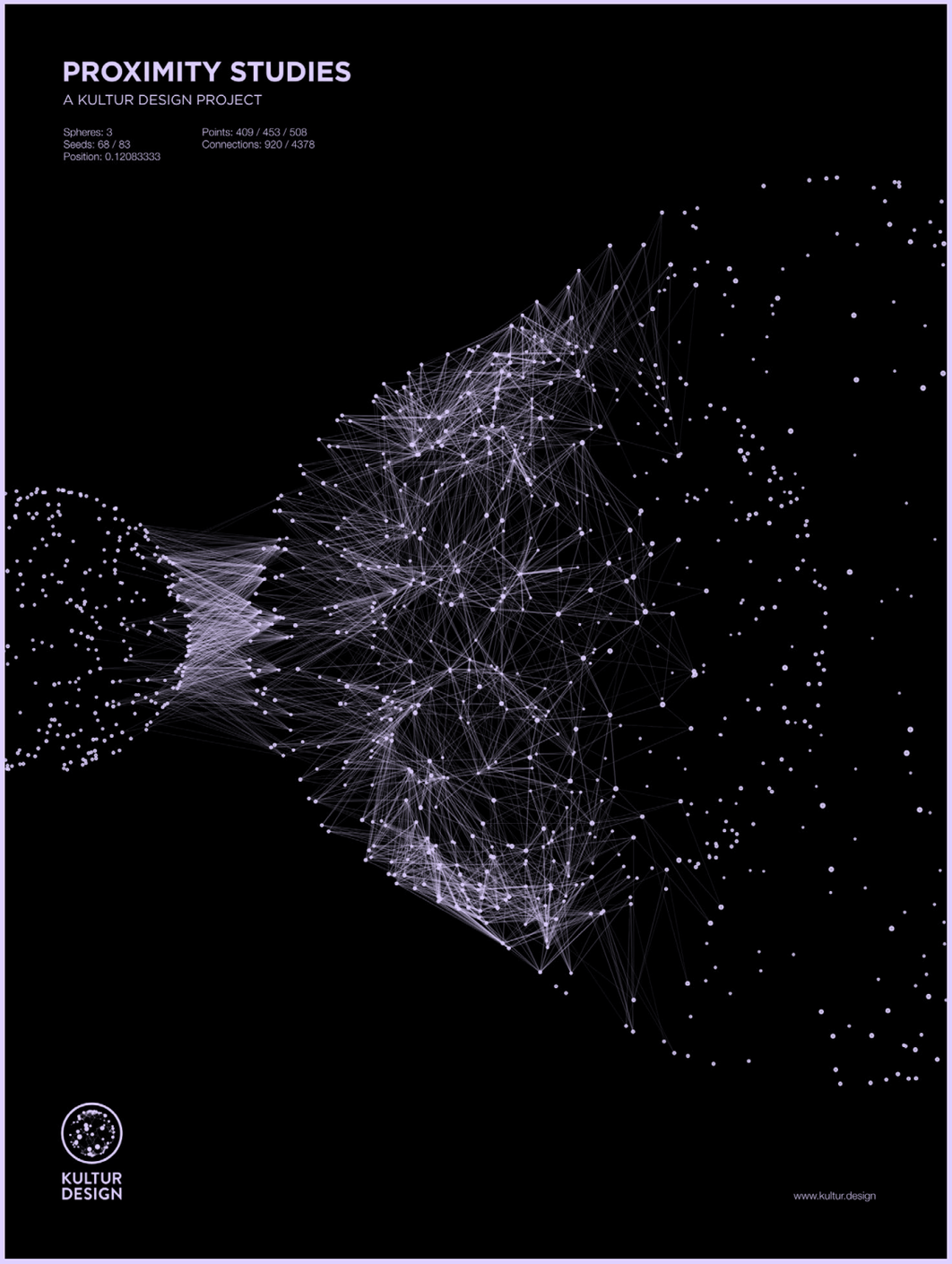


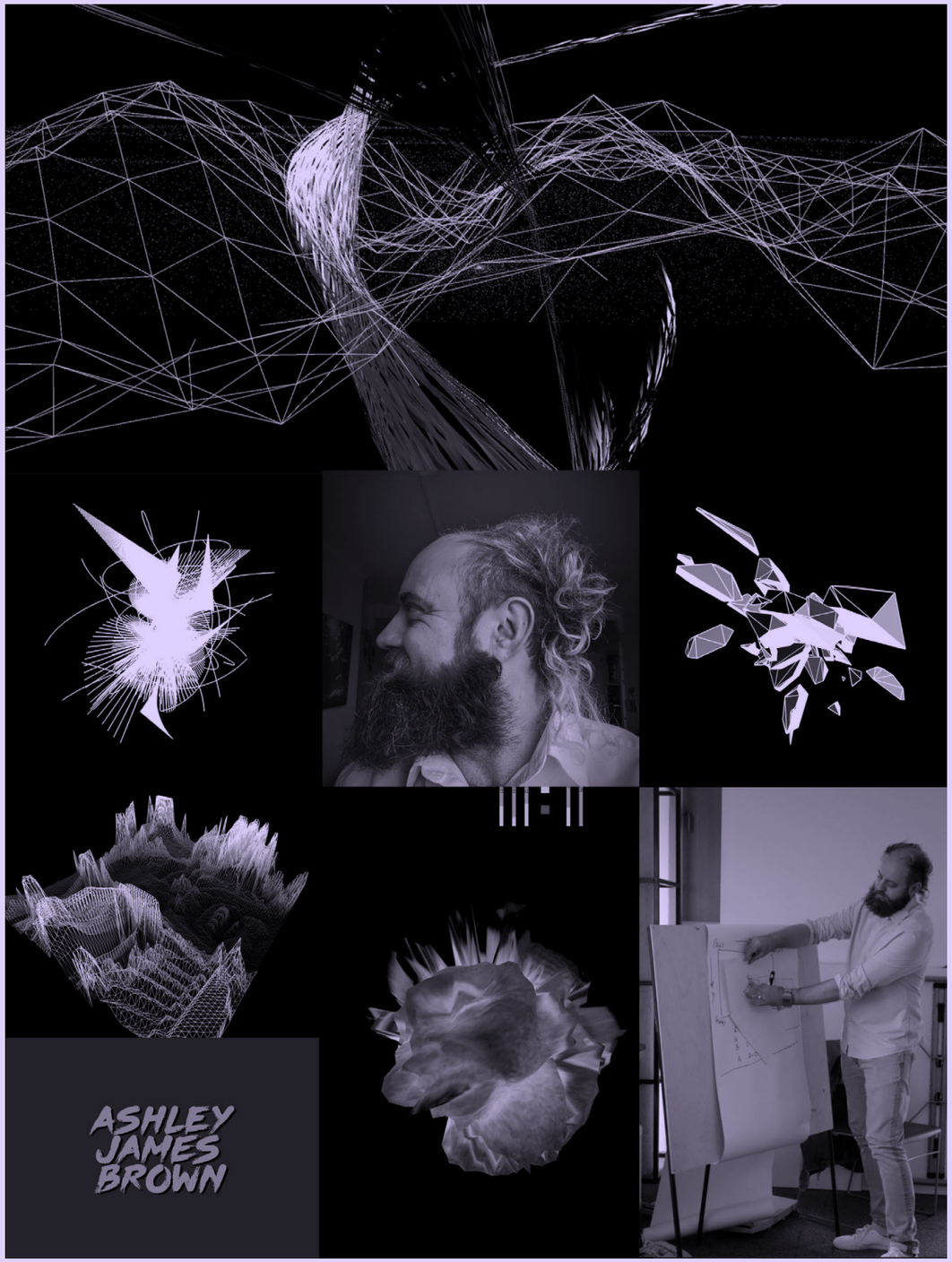








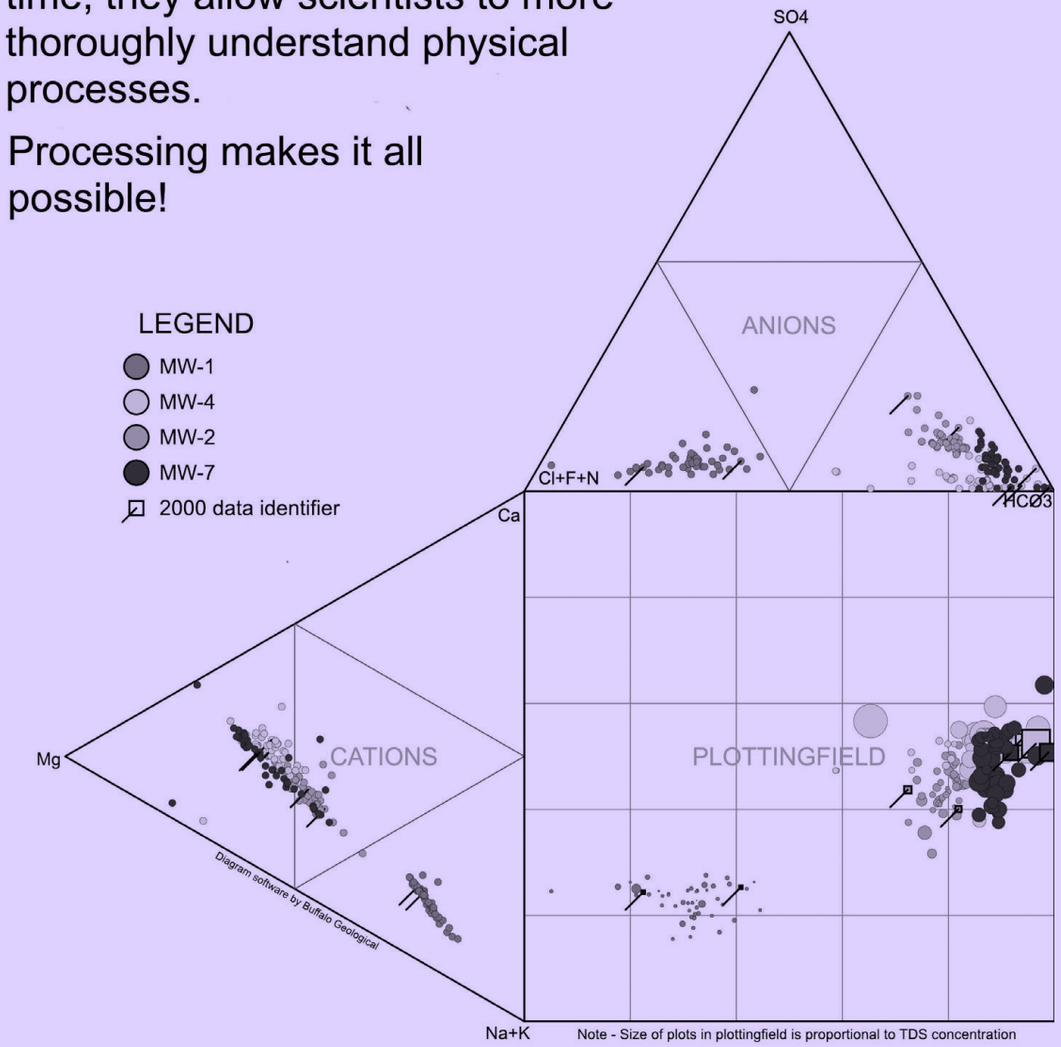

PROXIMITY STUDIES
A KULTUR DESIGN PROJECT
Spheres: 3
Seeds: 68 / 83
Position: 0.12083333
Points: 409 / 453 / 508
Connections: 920 / 4378
KULTUR
DESIGN
www.kultur.design





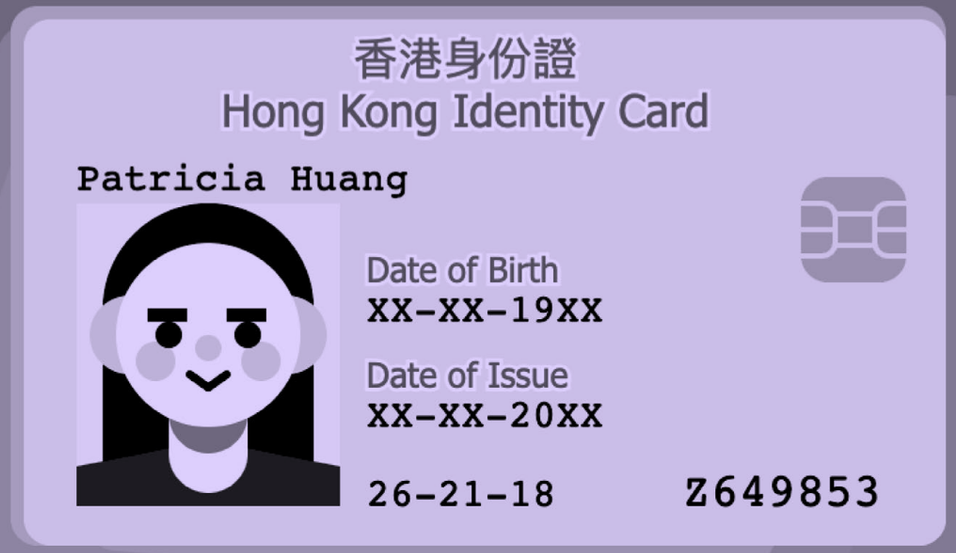

Water quality diagrams such as the Durov-type below illustrate the impact of contaminants on groundwater. When drawn in color and animated through time, they allow scientists to more thoroughly understand physical processes.

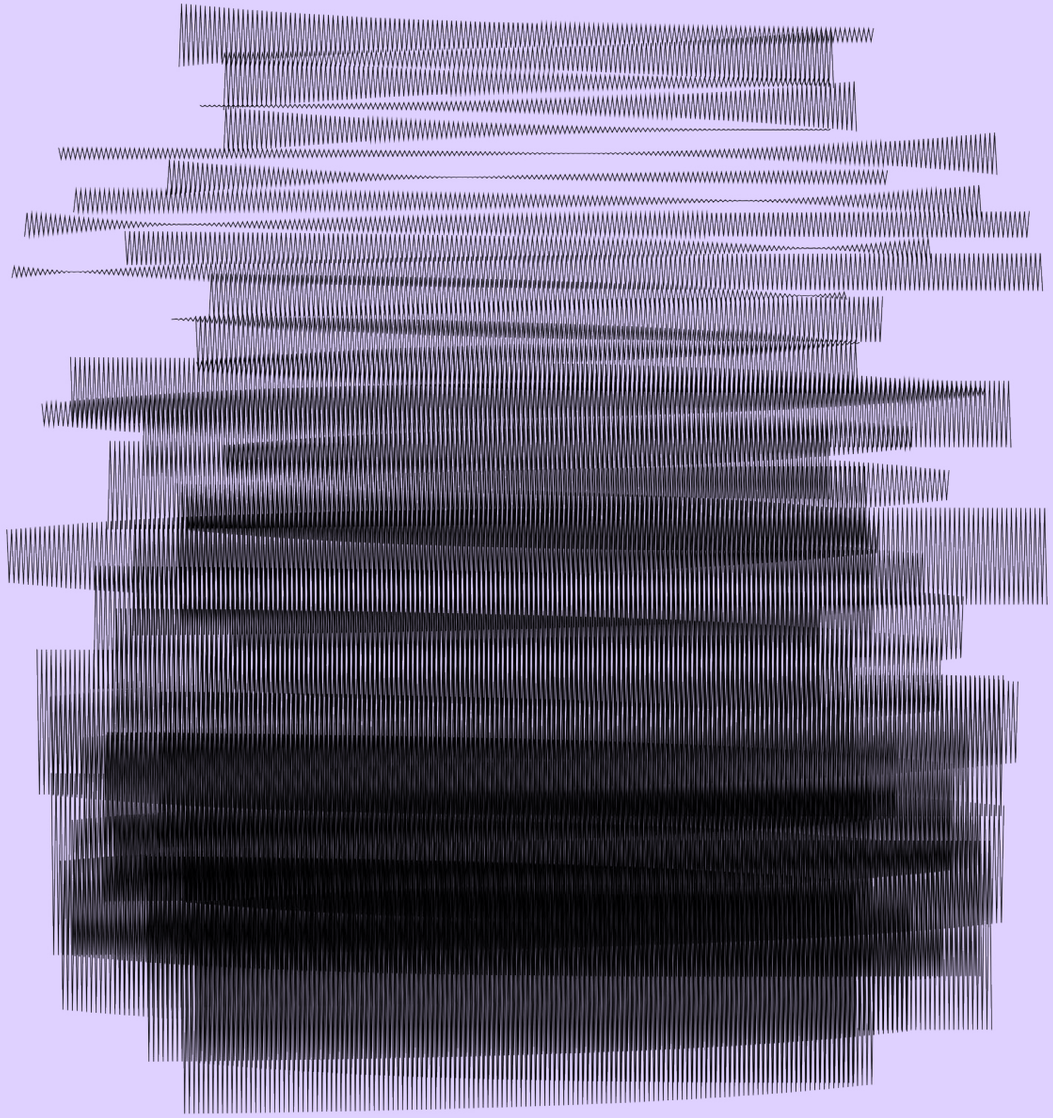
Processing makes it all possible!



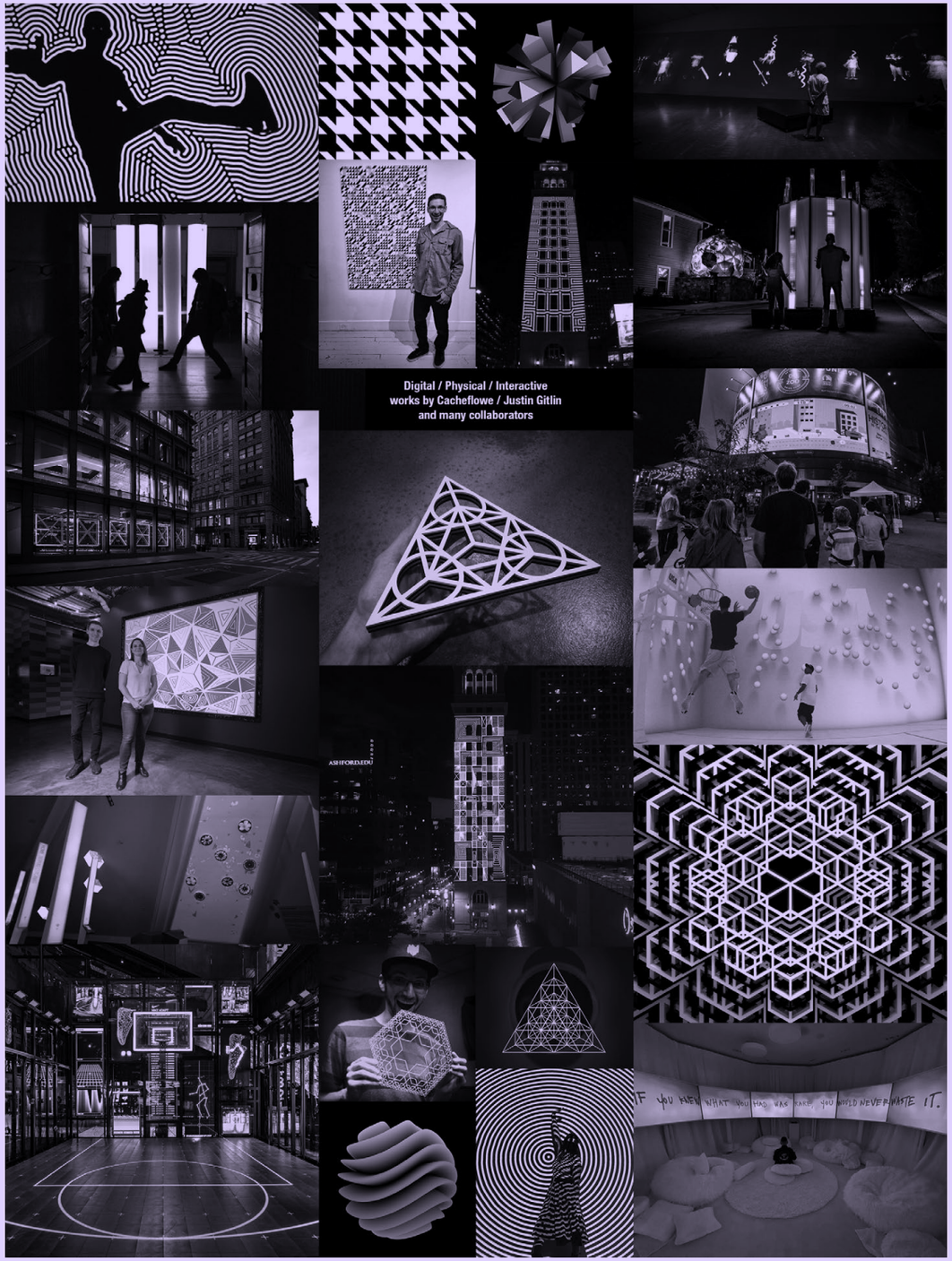
# Lost and Found

It is a regular rectangular card.
It has minimal color, mostly just gray.
It has a tiny chip on it. It has a name on it.
It has a date of birth on it and my photo.
It is issued by the Hong Kong government.
I used to be able to get into bars before I turned 21.



bustavo

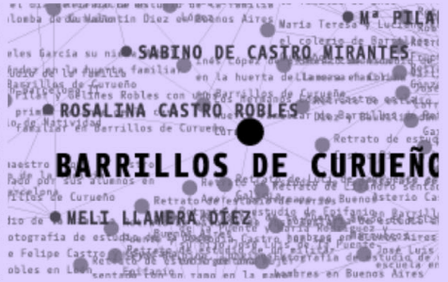



CACHEFLOWE / JUSTIN GITLIN

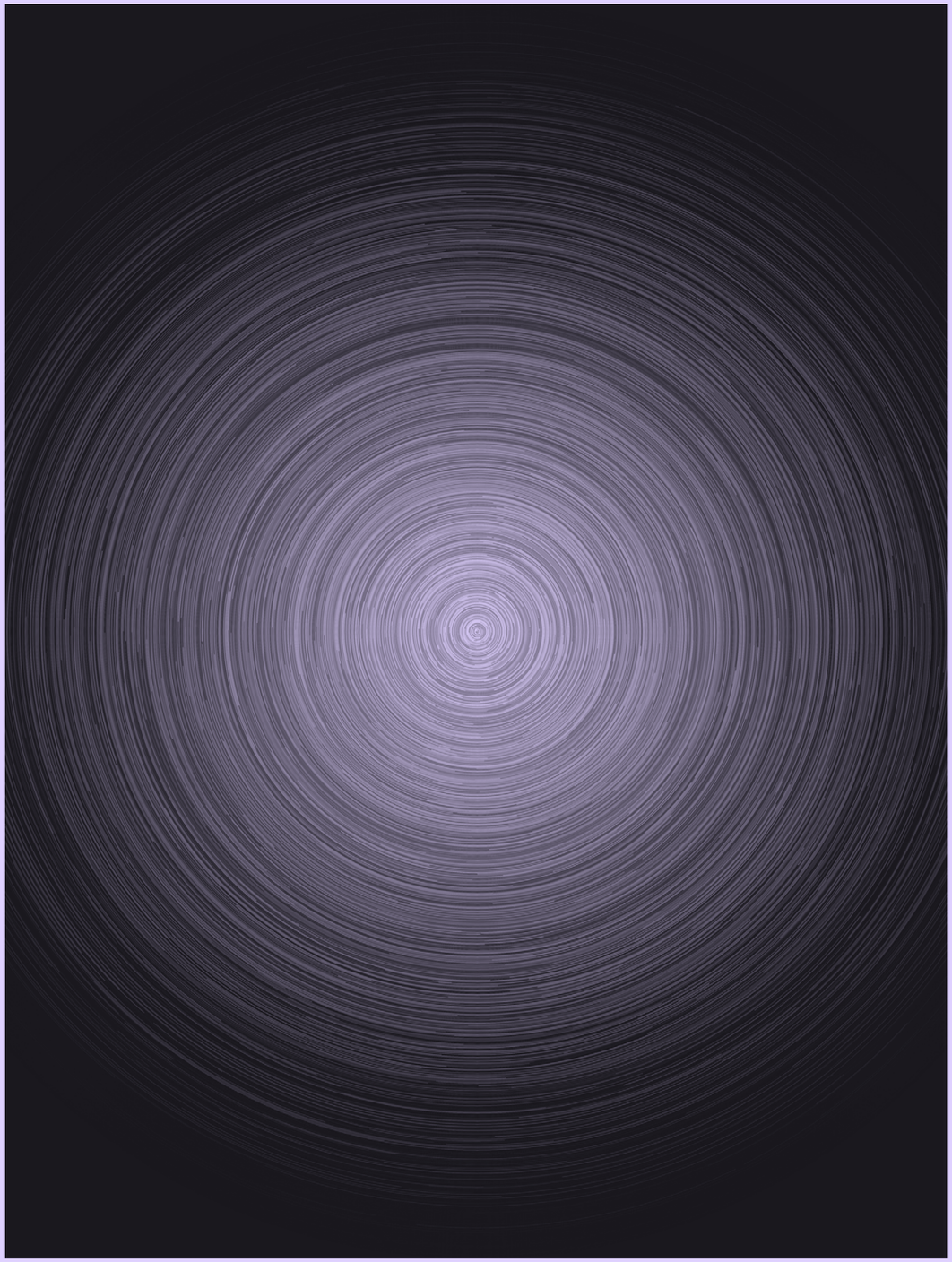



A 4 by 4 meters force directed diagram, part of “Territorio Archivo” (TA) a participatory promoted by Fundación Cerezales Antonino y Cinia in Cerezales del Condado – León (Spain), curated by Alfredo Puente and ideated by Chus Domínguez.

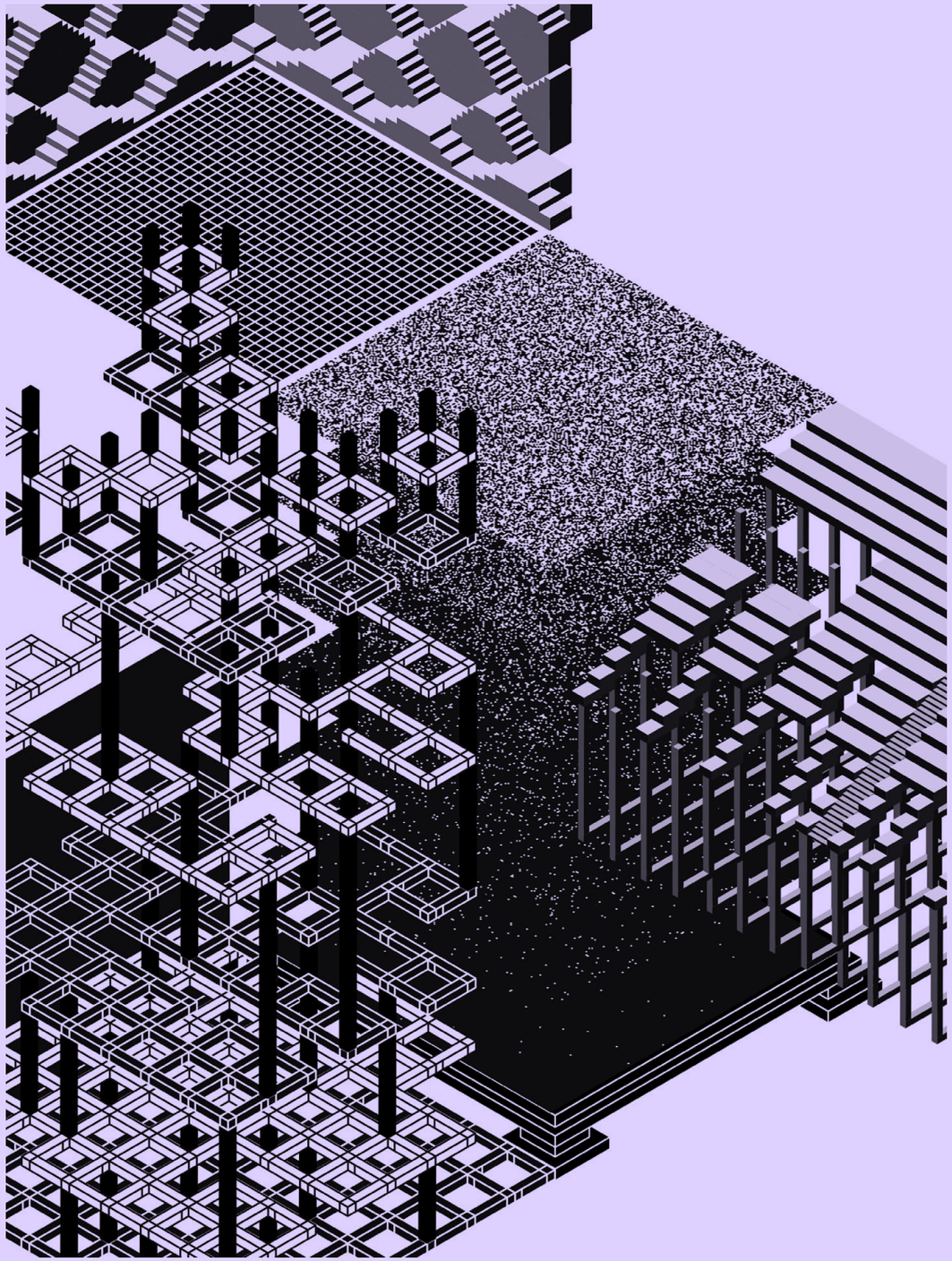
During TA, the inhabitants of six small villages in the region of Condado-Curueño were invited to share their family albums in a common archive and, by doing so, they became *domestic curators* of their pictures. This photographic archive was intended, since the beginning, as a starting trigger for an ongoing cycle of activities on the community’s memory.




ALFREDO CALOSCI - NEGOT.NET

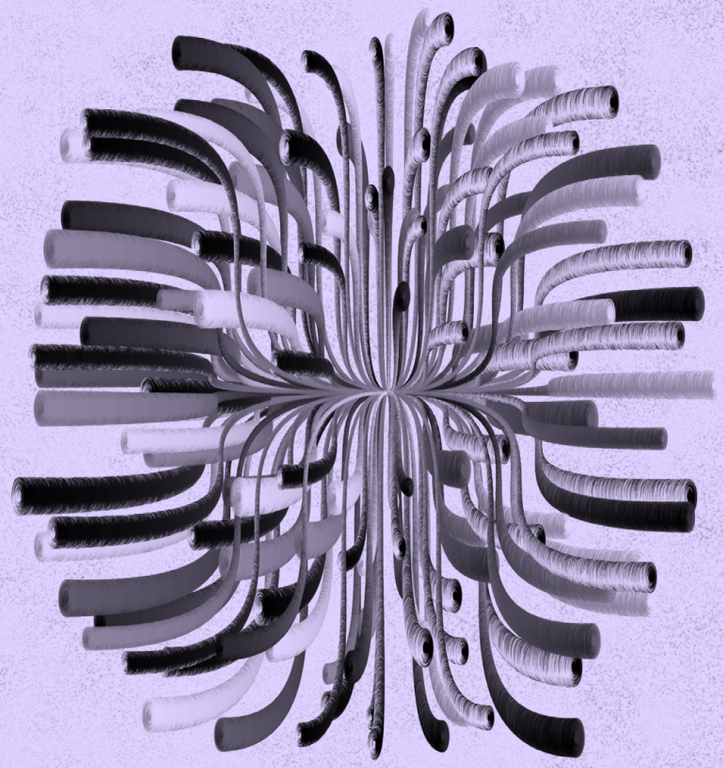






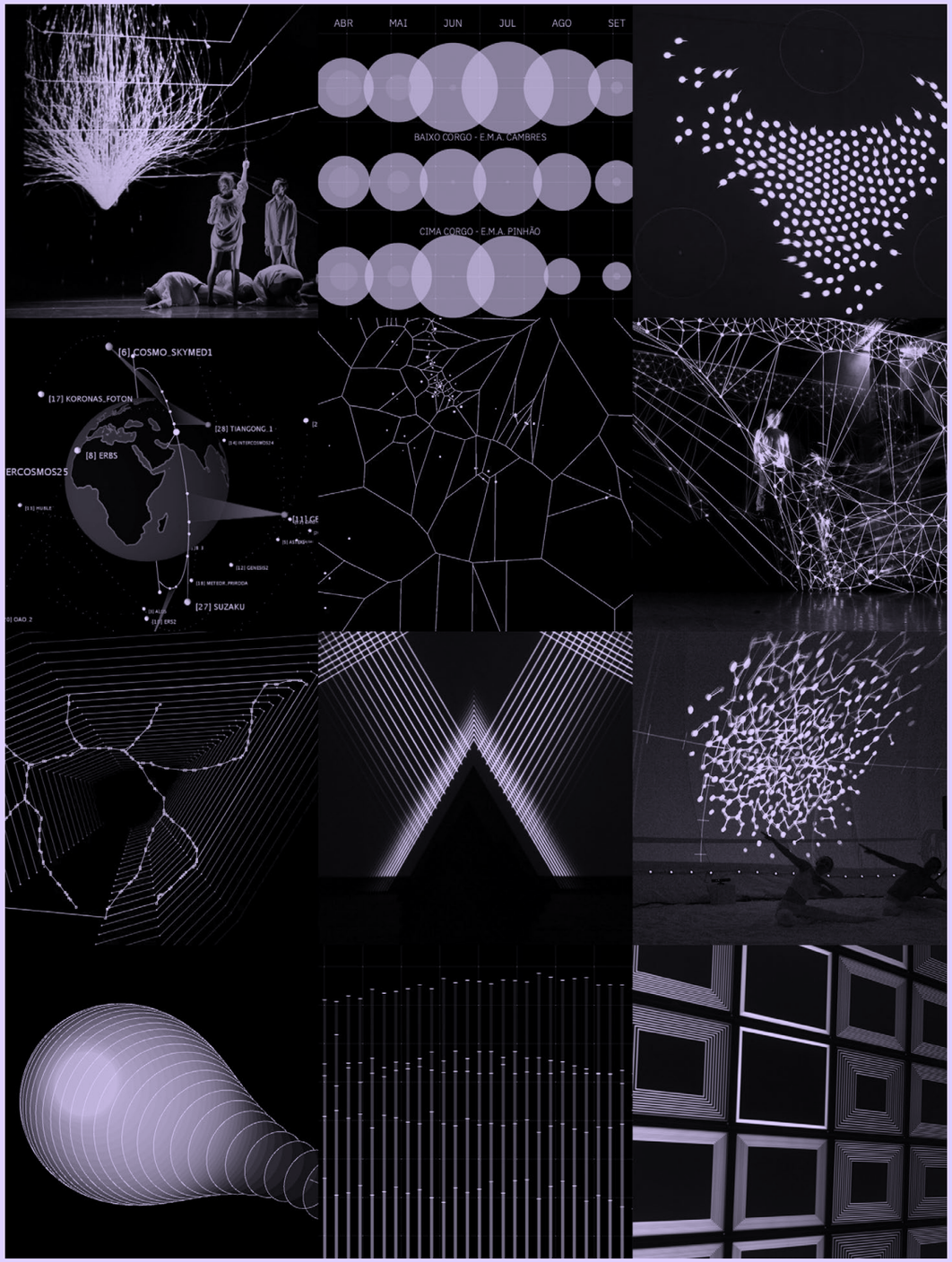




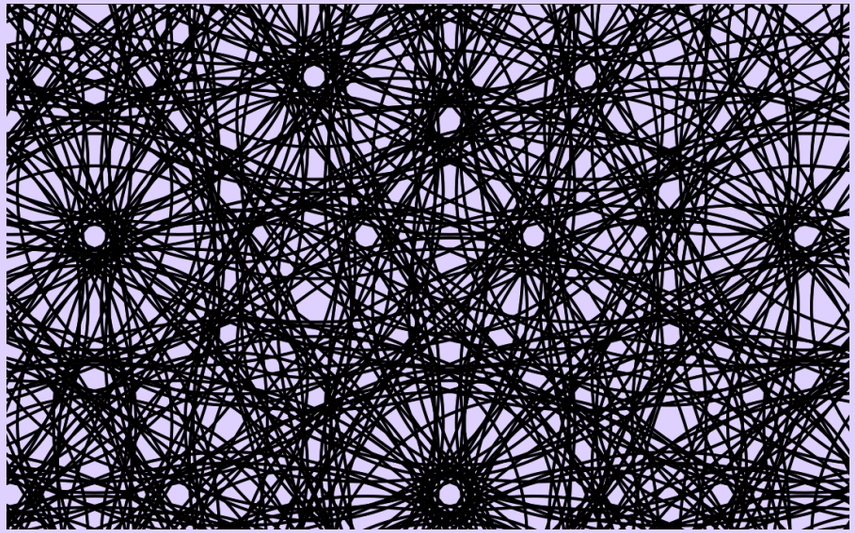


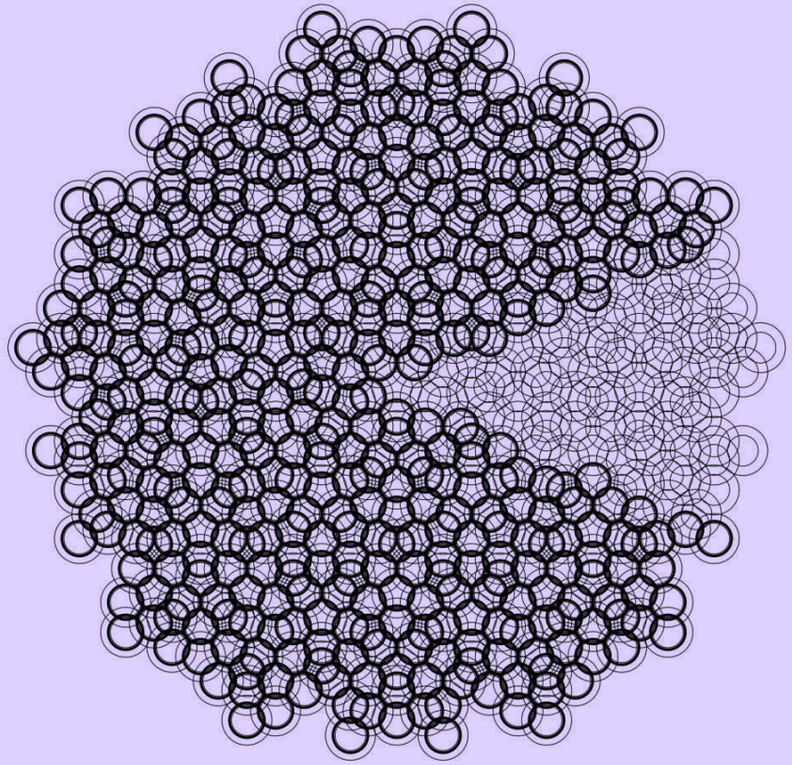
ǯixer créixer créixer créixer créixer

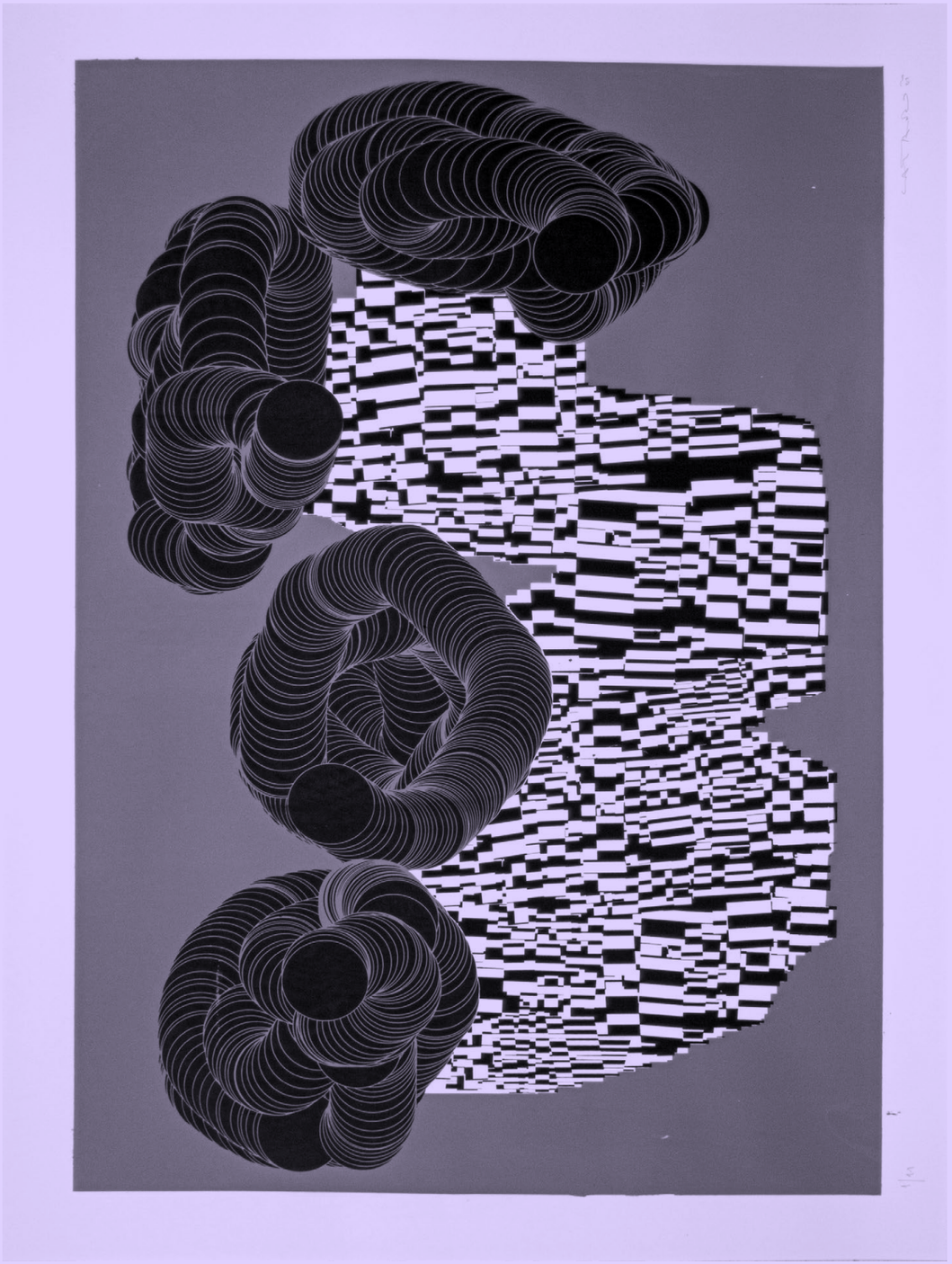
Néixer i créixer. Anna Carreras (2020)


ABR
MAI
JUN
JUL
AGO
SET
BAIXO CORGO - E.M.A. CAMBRES
CIMA CORGO - E.M.A. PINHÃO
[6] COSMO_SKYMED1
[17] KORONAS_FOTON
[28] TIANGONG_1
[8] ERBS
ERCOSMOS25
[27] SUZAKU

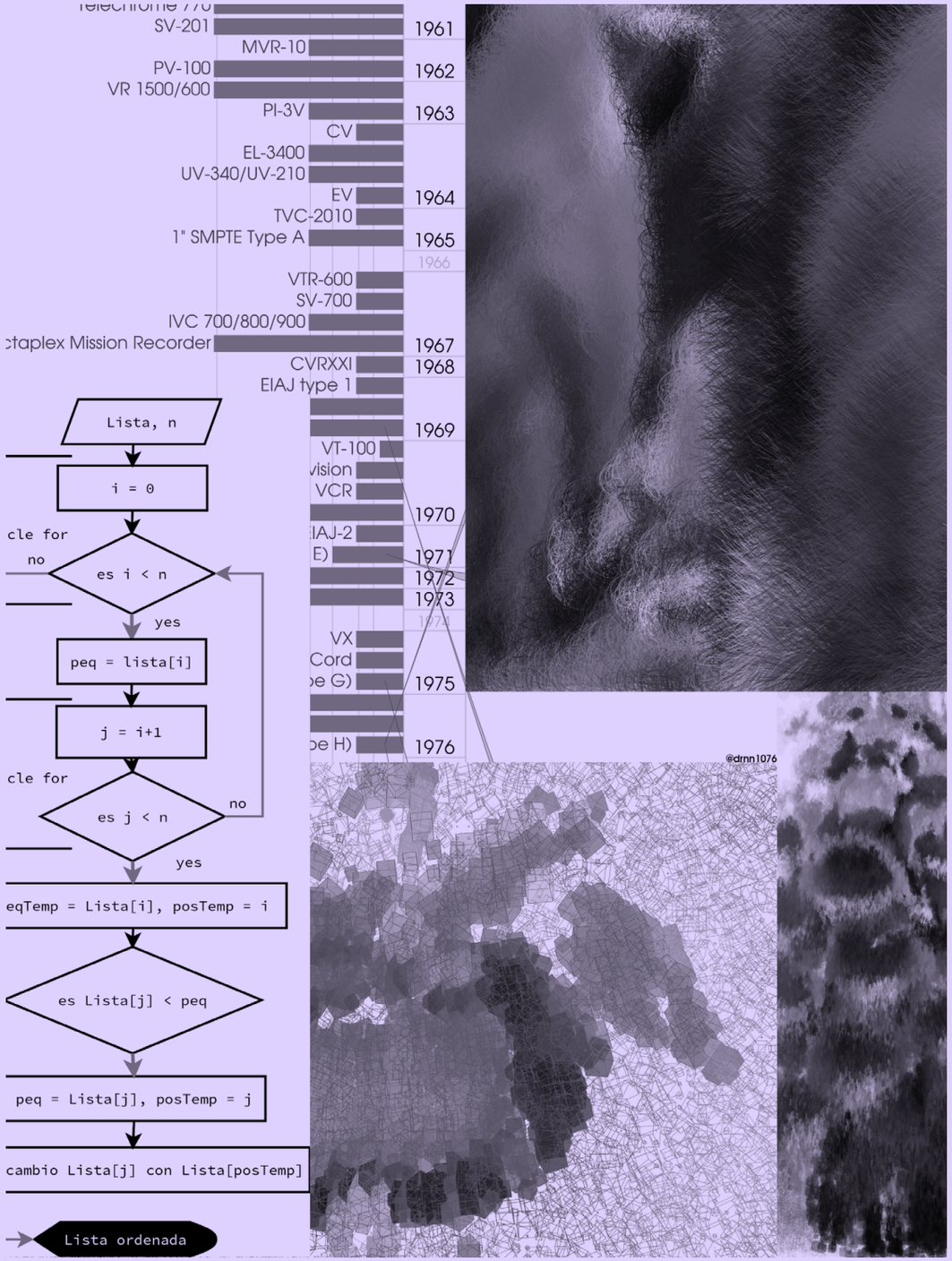











SV-201
MVR-10
PV-100
VR 1500/600
PI-3V
CV
EL-3400
UV-340/UV-210
EV
TVC-2010
1" SMPTE Type A
VTR-600
SV-700
IVC 700/800/900
CVRXXI
EIAJ type 1
VT-100
VCR
VX
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1975
1976
Lista, n
i = 0
es i < n
yes
no
peq = lista[i]
j = i+1
es j < n
eqTemp = Lista[i], posTemp = i
es Lista[j] < peq
peq = Lista[j], posTemp = j
cambio Lista[j] con Lista[posTemp]
Lista ordenada
@drnn1076

Nearly 10 years ago, I had just started my journey in programming, and came across Daniel Shiffman's The Nature of Code. I quickly became addicted to the visual feedback of Processing and began my first wobbly steps of building programs (bouncing balls, Snake, raindrop simulation, etc.) within the friendly environment of the Processing editor.

Little did I know that in the next decade, Processing would not only be the start of a (probably lifelong) journey into tech and interactive art, but would also literally take me across the world.

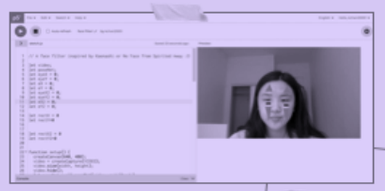
Since my first Sketch, Processing has given me an internship, a class to teach, a music video, my first Developer Conference, and countless nights of coding, debugging, and exhilaration when ideas turn to reality.

If you are reading this, thank you for being a part of this community! I hope you build with a playful spirit, joy, and with plenty of friends!




# me and p5.js and processing

```
void setup() {
  console.log ("my first
  sketches in
  processing");
};
```



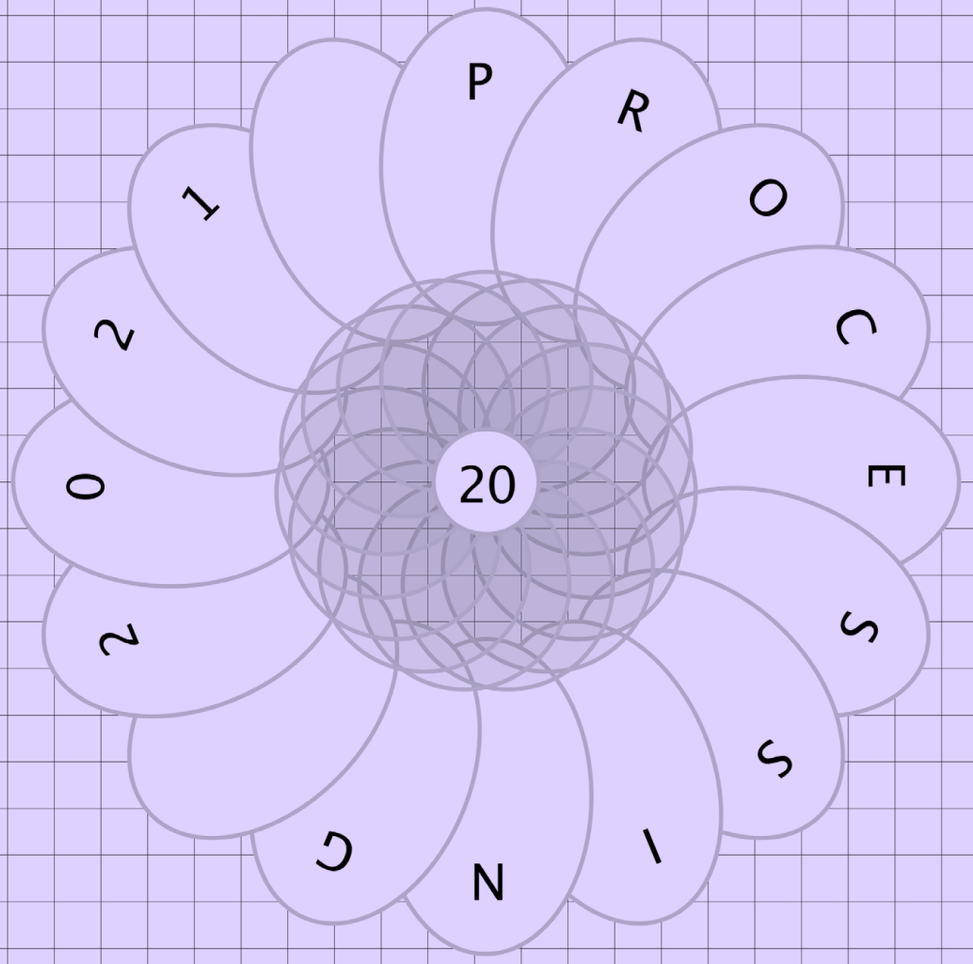
```
function draw() {
  console.log ("
  trying new
  libraries with
  p5.js like PoseNet
  and Glitch!");
};
```



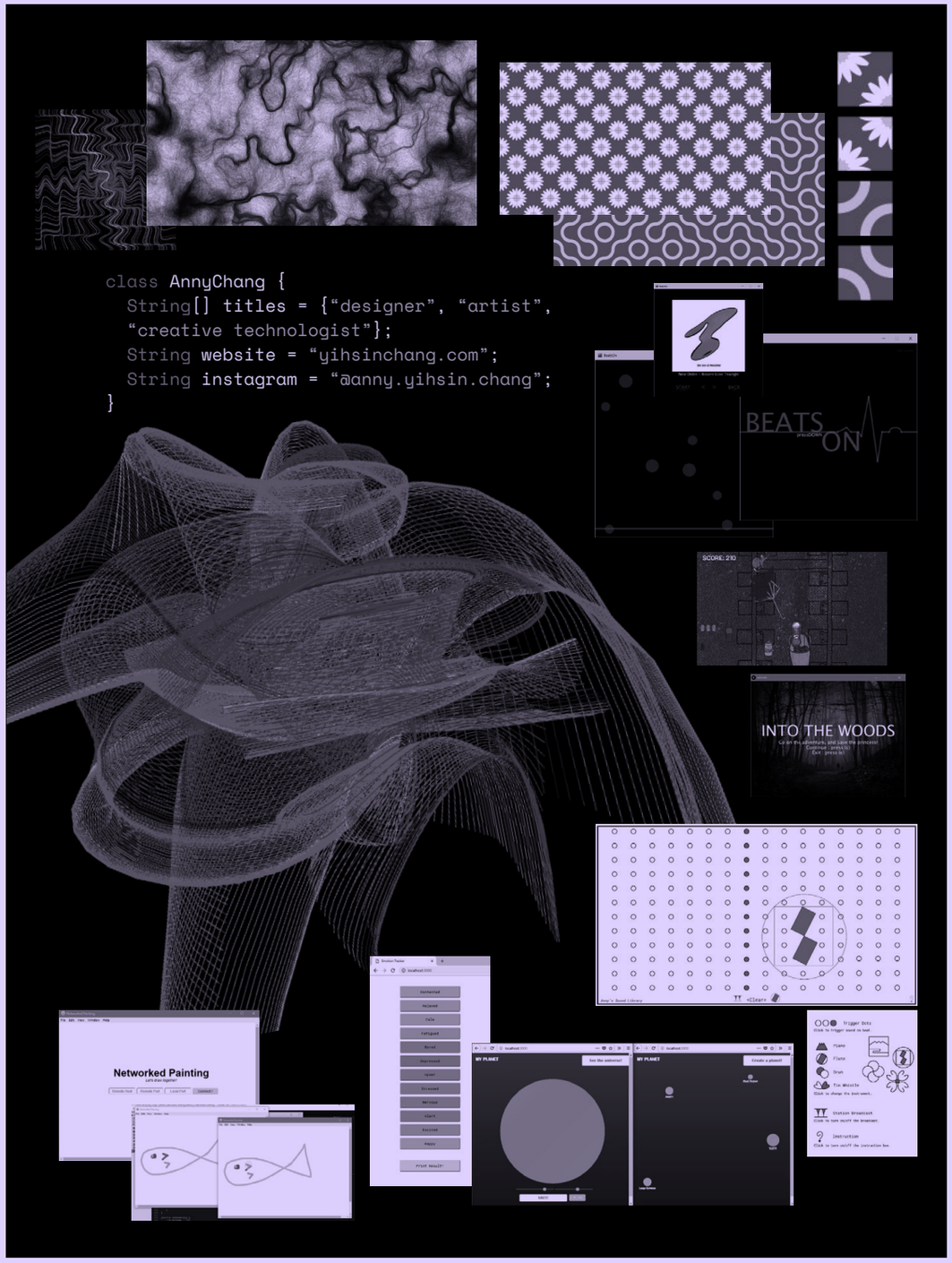
```
function GSoC21 (p5jsshowcase) {
  in the summer of 2021, i had
  the chance to continue the
  development of the p5.js showcase,
  specifically curating it by theme.
  it was one of the most special and
  rewarding experiences and i was
  enamored by the p5.js community and
  their passion for pushing the
  perceptions of how p5.js can be used.
  shoutout to my mentors ashley kang,
  kt duffy, and sam vassor <3
};
```

view at showcase.p5js.org

katie chan, 2021


PROCESS
2021
20



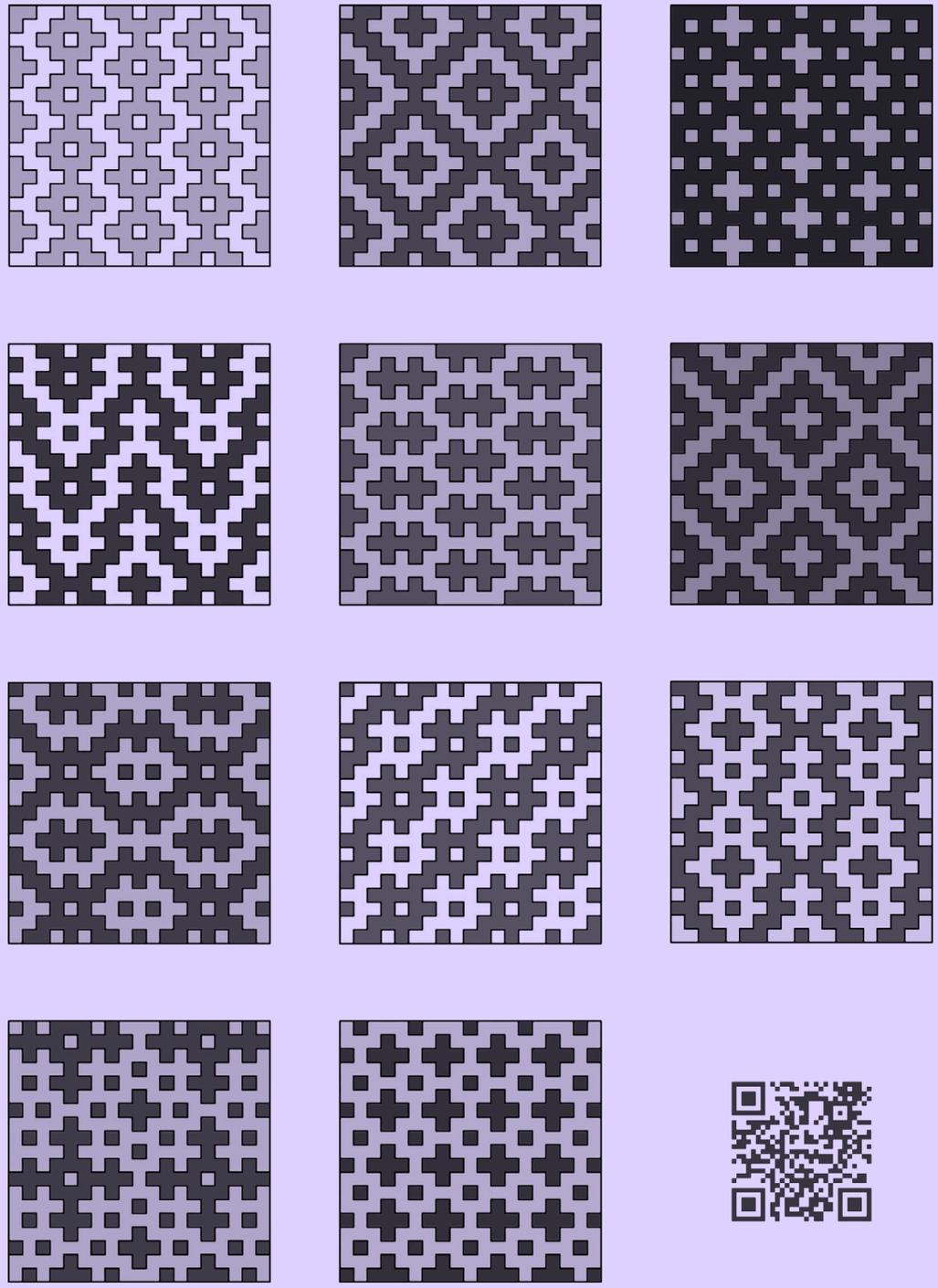

class AnnyChang {
String[] titles = {“designer”, “artist”,
“creative technologist”};
String website = “yihsinchang.com”;
String instagram = “@anny.yihsin.chang”;
}
BEATS ON
SCORE: 210
INTO THE WOODS
Networked Painting

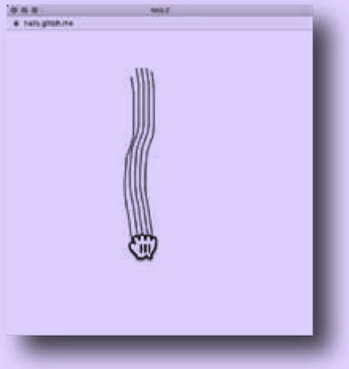
# MANIFEST

was a project by **Michael Chang** while studying at *UCLA Design | Media Arts* with Casey Reas' "Programming Media II". It was a biologically inspired generative art piece which let you draw lines that transformed into swimming organism, living in a fluid-like environment.

Manifest was one of the first Processing sketches to utilize vector art, utilizing a custom SVG Parser. This parser became an early implementation of Processing's SVG Support.



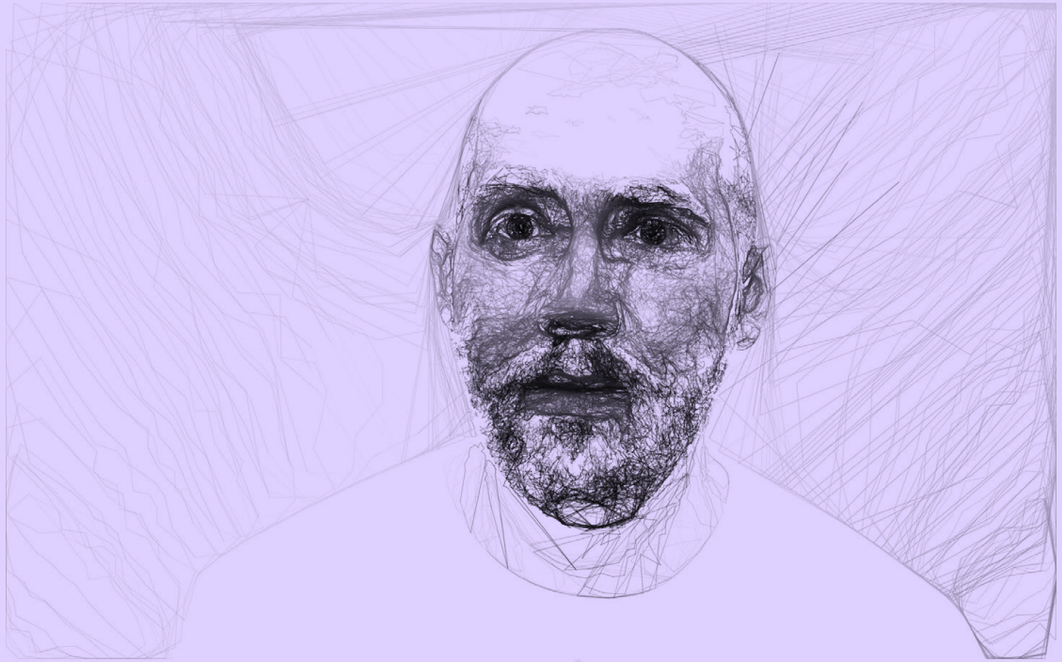






# Project #3 experimental clock



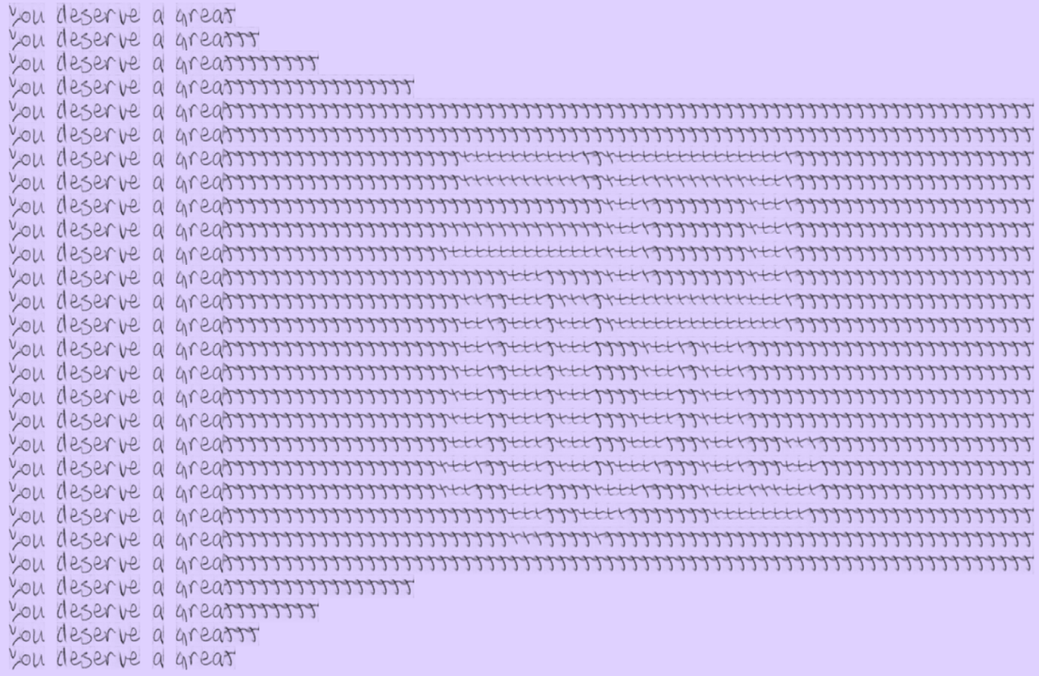



In this interactive application, the user can selectively spawn autonomous agents on the canvas using the mouse. There is a small collection of different sets of rules to choose from and change the behavior of the agents that trace and recreate an underlying photo in various styles.



Dear Processing

HaPPy twenty years
Here is a Hooray for every month you helped make
Software and art more accessible


Sincerely Yours

PAST
PRESENT
FUTURE
17:31:29

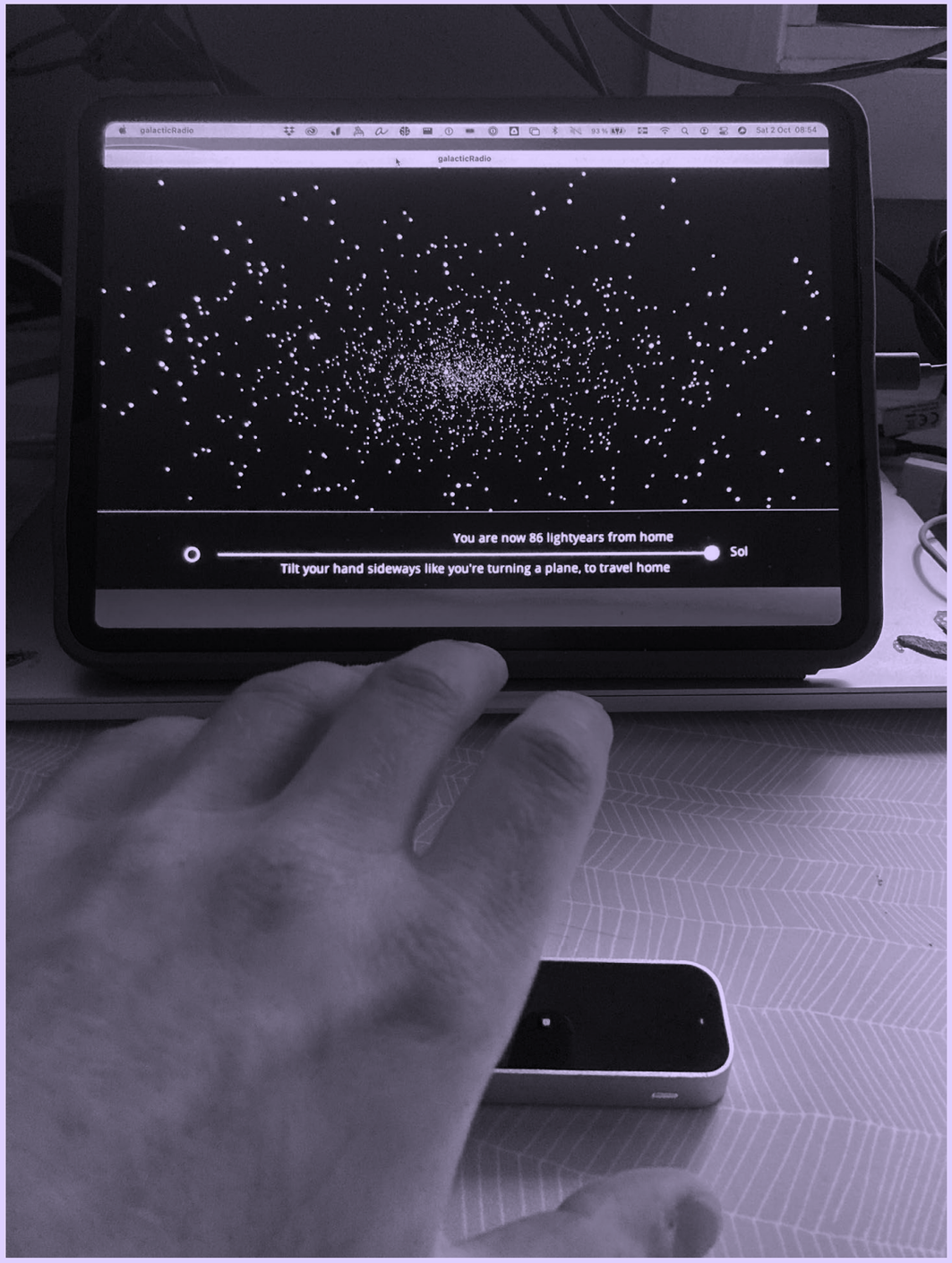
PAST
PRESENT
FUTURE
17:31:27

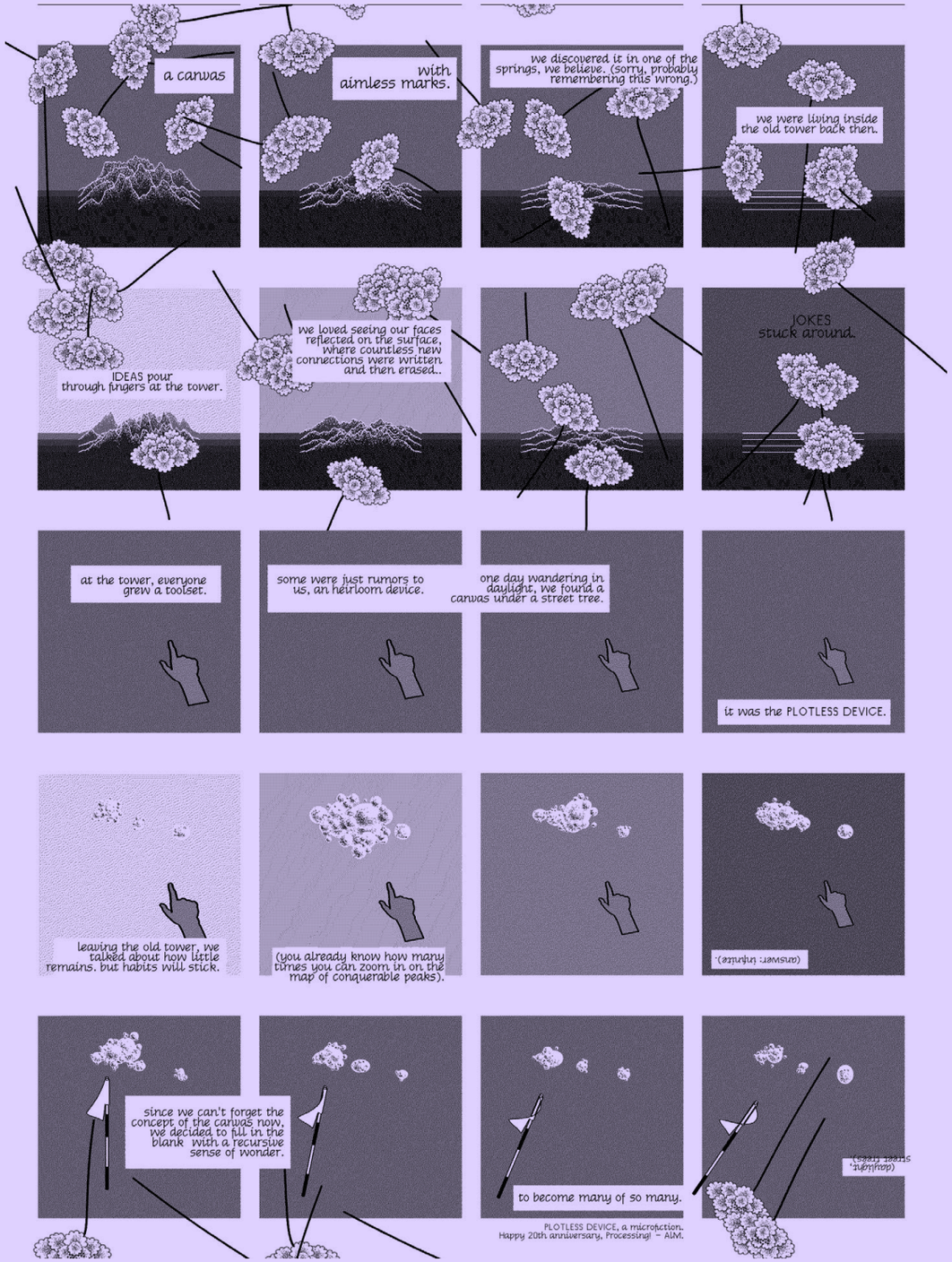
PAST
PRESENT
FUTURE
17:31:28

## HARMONY CLOCK

This clock is gives three options for time viewing; past, present, and future. It uses the hover function to display the time at present, -1s for the time past, and +1s for the time future. For past and future the time is displayed atop the present time. Without hovering the present time is displayed. This clock is a reminder to "live in the present" as that is the only way harmony is acheived.

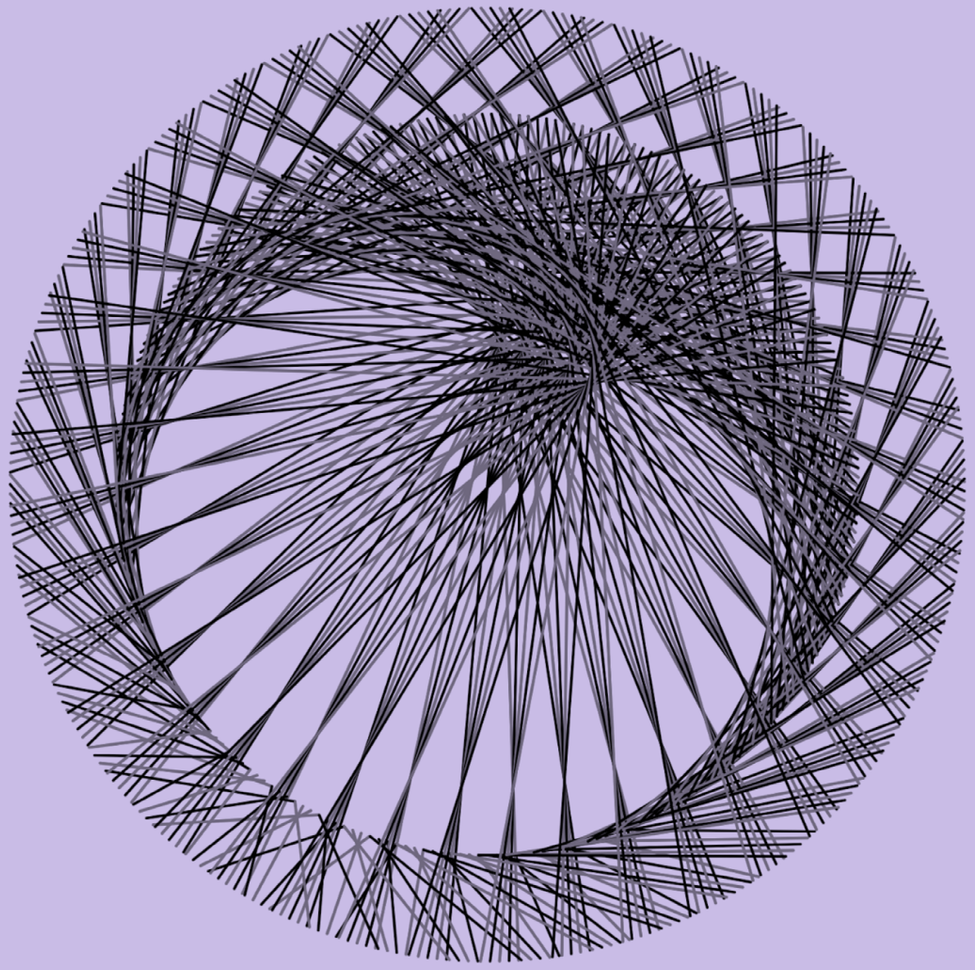
```
let h = hour();
let m = minute();
let s = second();

text(h+":"+m+":"+s, 183,333)

//present
  if (mouseX > 50 && mouseY > 145
&& mouseX < 360 && mouseY <  200) {
  fill(255,255,0);
  textSize(75);
  text('PRESENT', width/2,height/2);
  text(h+":"+m+":"+s, 183,333);
  }

//past
  else if (mouseX > 50 && mouseY > 75
&& mouseX <220 && mouseY  < 132)  {
  fill(255,255,0);
  textSize(75);
  text('PAST', 400/3,400/3);
  text(h+":"+m+":"+(s-1), 183, 333);
  }

//future
  else if (mouseX > 50 && mouseY > 210
&& mouseX <310 && mouseY < 264) {
  fill(255,255,0);
  textSize(75);
  text('FUTURE',400/2.25,400/1.5);
  text(h+":"+m+":"+(s+1), 183, 333);
  }
```



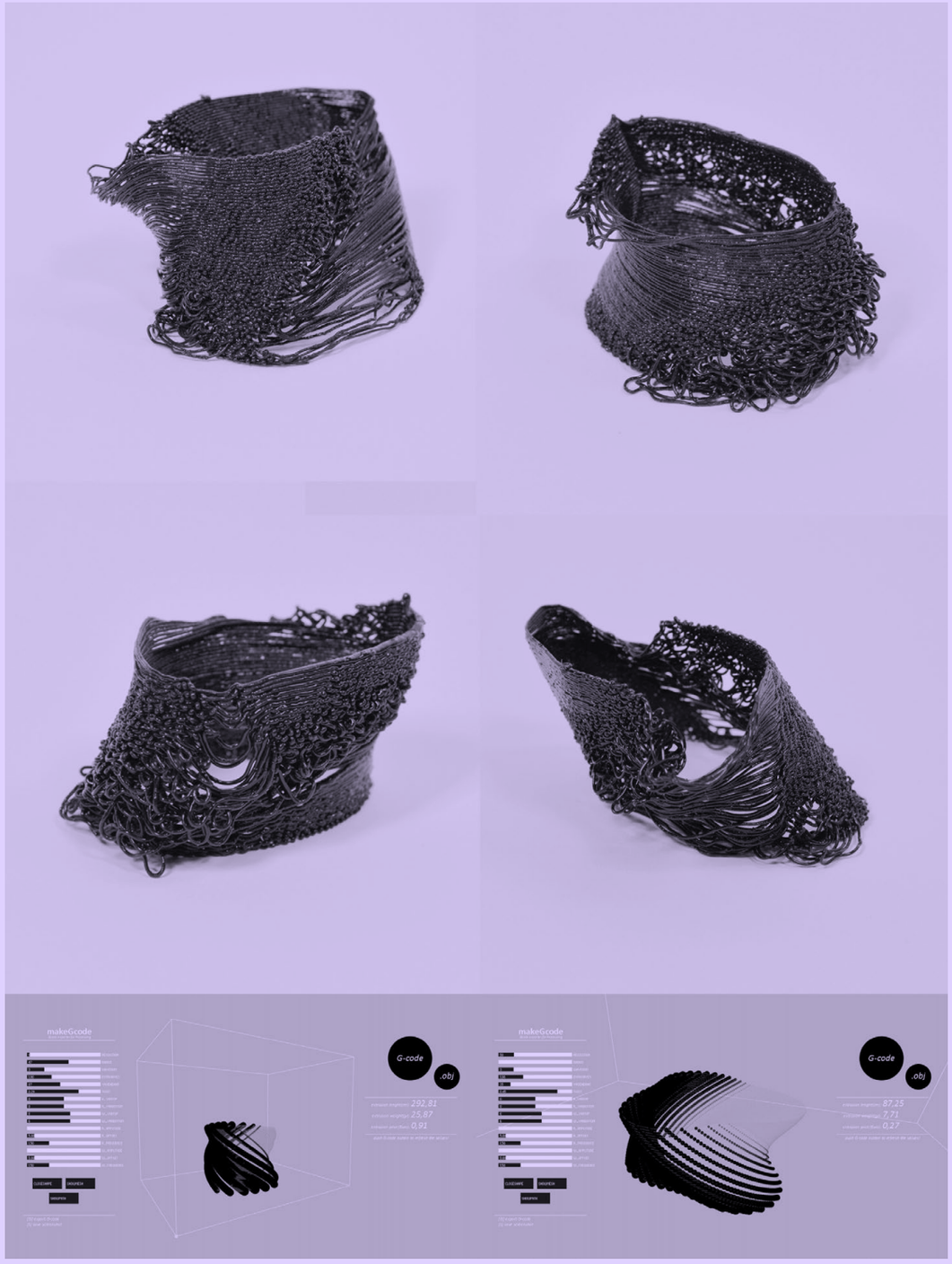

a canvas
with aimless marks.
we discovered it in one of the springs, we believe. (sorry, probably remembering this wrong.)
we were living inside the old tower back then.
IDEAS pour through fingers at the tower.
we loved seeing our faces reflected on the surface, where countless new connections were written and then erased..
JOKES stuck around.
at the tower, everyone grew a toolset.
some were just rumors to us, an heirloom device.
one day wandering in daylight, we found a canvas under a street tree.
it was the PLOTLESS DEVICE.
leaving the old tower, we talked about how little remains. but habits will stick.
(you already know how many times you can zoom in on the map of conquerable peaks).
(answer: infinite).
since we can't forget the concept of the canvas now, we decided to fill in the blank with a recursive sense of wonder.
to become many of so many.
PLOTLESS DEVICE, a microfiction.
Happy 20th anniversary, Processing! – ALM.




Sacred






makeGcode
G-code
.obj
makeGcode
G-code
.obj




```
//www.pcdaarhus.net > why code? (open your web browser console)
let celebrate_PCD = [
"Ester Marie Aagard (2019)\
    -I want to understand the hidden parts as data-tracking, the way in which different software's are programmed and
    so forth. [...]I use coding both to understand and explain the software that we use",
"Andreja Andric (2019)\
    -I subscribe to the Pythagorean idea that numbers are behind everything. With this in mind, coding helps me
    connect to the hidden side of the world and is a form of contemplative action",
"Anna Brynskov (2020)\
    -With knowledge about coding you can change the world. Or at least you can have a stronger voice in discussions
    about how society should be built and designed. Knowledge is power",
"Tilde Lageri Damborg (2019)\
    -Coding to me is not only a tool to reach a end goal, its an inspiring, strict, rigid and sometimes very annoying
    co-designer",
"Raune Frankjær (2021) \
    -To me it's almost like some a kind of modern day shamanism - using sensing technology [...] to reveal the
    forgotten yet amazingly vibrant and intertwined multispecies world that we live in and are part of",
"Rolf Holm (2020)\
    -Personally for me, coding is fun. Someone once called it, 'the greatest puzzle', and I get why. It's the most
    open-ended game there is, with no end goal beyond creativity and mastery.",
"Niels Konrad (2020)\
    -When I started studying deisgn, I remember I saw coding and prototyping with code as a little spark, a little bit
    of magic that could breathe life into already appealing visual design.",
"Simon Feusi Ludvigsen(2021)\
    -Knowledge about coding gives you the tools to change the world in your own way. It gives you the ability to solve
    problems from a new perspective that the majority of people don't have knowledge about.",
"Alex Mørch (2020)\
    -Besides being a great compositorial tool for autonomous and interactive music, coding also bridges digital and
    physical space",
"Gabriel Pereira (2020)\
    -I think I'm interested in coding because I'm ultimately afraid of it. [...]I (as anyone), am accountable to it,
    and to building it into something that supports and advances human rights, and social/economic justice",
"Magda Tyżlik-Carver (2020)\
    -code / language, code / translation, computation before code, worlding practices",
"Kristoffer Ørum (2021)\
    -Why would I not code? [...] I mostly use code to break or hack technological artifacts such as routers or
    screens, in order to make them do things they were not originally intended to do.",
"Mark Staun Poulsen (2019)\
    -code underwrites so many aspects of contemporary digital living[...]I have a unique chance to explore meaning and
    consequences of computation in light of a creative and practical engagement with programming",
"Mace Ojala (2021)\
    -Basically, to code is to seek a lasting relationship which is based not only on one-sided use, but on mutual
    transformation. [...]that is a beautiful, worthwhile and even rebellious thing to do.",
"Sofie Fürsterling Mønster (2021)\
    -what I find most intriguing about coding is how I have the possibility to increase awareness and critical
    reflection about topics that I find important to draw attention to, especially as a lack of critical reflection
    helps to perpetuate this inequality…",
"Lasse Korsgaard (2020)\
    -The main reason I code and love coding is that I have the ability to create my own tools to use in my creative
    process",
"Ann Karring (2019)\
    -To me coding is a way to express myself. Whether it is personal or global issues, coding helps me reflect on
    these issues. I also use it as a tool of communication, so I can share my views with others",
"Malthe Stavning Erslev (2019)\
    -I am interested in coding because it inspires me and enables me to think trough conceptual ideas, notions,
    concerns etc. I will often start out with a vague idea of a concept that I want to explore, and the practice of
    coding will 'take' me somewhere I did not anticipate.",
"Nanna Debois Buhl (2021)\
    -What happens when the weaving-computing relationship is examined from a bodily and material perspective and when
    we engage the loom as a computer, as a time machine?[...]I explore the connections between the threads of the loom
    and the programming of the computer",
"Nina Isis Kinch Bolton (2021)\
    -I believe it is important to understand one of the most prevalent creation tools of today so that we can question
    and create for all and not just the privileged few",
"Sofie Andersen (2021)\
    -Coding for me, is a way of expressing yourself just like art and literature. Just like traditional artforms have
    been used to raise critique and awareness of important matters, I see coding as the new artform that gives
    empowerment back to the individual."
];

function draw() {
    frameRate(1);
    print(random(celebrate_PCD));
}
```


Creative coding class at Hof University / Campus Muenchberg / Communication Design
Portrait remixes 2021 – p5js / OpenProcessing – Prof. Michael Zoellner
https://openprocessing.org/class/68512

p5.js shaders

Home GitHub Glitch Collection

**Getting started**

- Introduction to p5.js shaders
- Why use shaders?
- What are shaders?
- Setting up shaders in p5.js
- How to write a shader
- Important shader concepts
- Using Variable Qualifiers
- Using Texture Coordinates

**Basic Examples**

- Basic gradient shader
- Gradient shader using texcoord
- Uniforms (Interactivity)
- Applying Shaders as Texture
- Applying Shaders to Vertices
- Texture Coordinates

**Advanced Examples**

- Image Effects (Webcam)

**Experimental Examples**

- Bodypix Feedback

**Working with other resources**

- Porting Shadertoy Examples
- Book of Shaders with p5.js

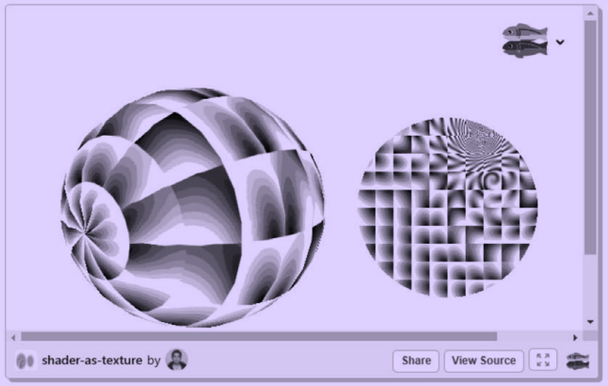
**Learn more**

- Check out these resources

## Welcome to p5.js shaders

Shaders are a great way to make graphics for your projects. So far you have probably made graphic shapes like **rect()** or **ellipse**() or maybe even 3D shapes like a **plane**() or a **box**(), and you should be more than familiar with using **fill()** to decide the color of these objects. But wouldn't it be great if we could make these objects have moving graphics on them? And without slowing down our sketch?

Enter the shader!




In this guide you will learn what a shader is, how to use it efficiently in p5, and how to make your own!

Consider this "the booklet of shaders". A great way to start familiarizing yourself with making amazing graphics for your projects in p5 and a simple introduction to shaders. Then, if you find yourself entranced by the world of shaders (like we do) please see the section on moving further with shaders, where we have collected some of the best sources online for finding shaders and learning more.

One of these, "The Book of Shaders" by Patricio Gonzalez Vivo and Jen Lowe, is a truly great resource if you want to dive deeper into the technical details of shaders, and we will be borrowing some explanations from this online book.

### Thank you

This guide is an open-source project made during xStory at ITP NYU by Casey Conchinha and Louise Lessél.

We want to thank the following people for help and guidance during the project:

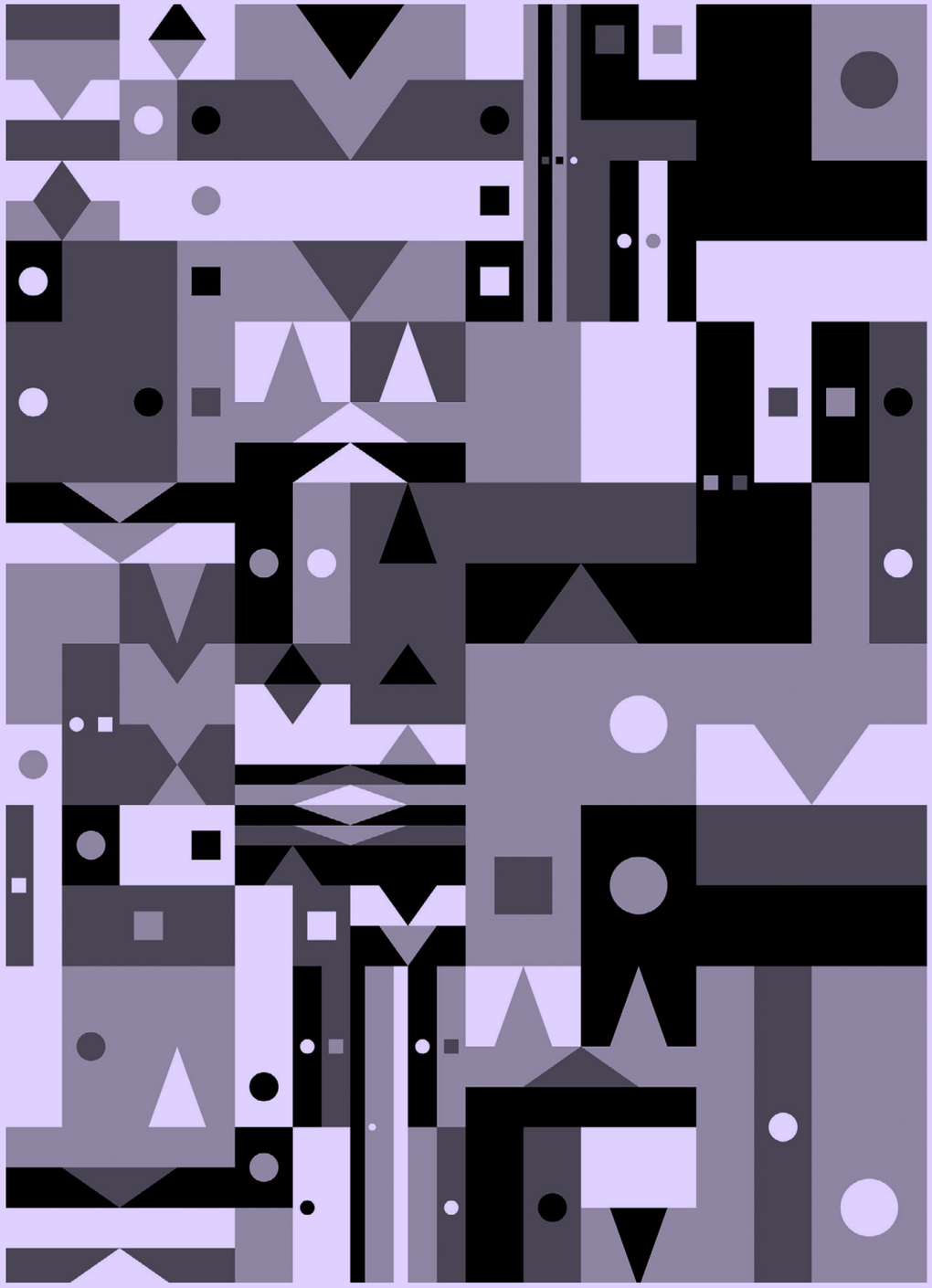
Adam Ferriss, for laying the groundwork for the entire project with his own p5 shader github repo.

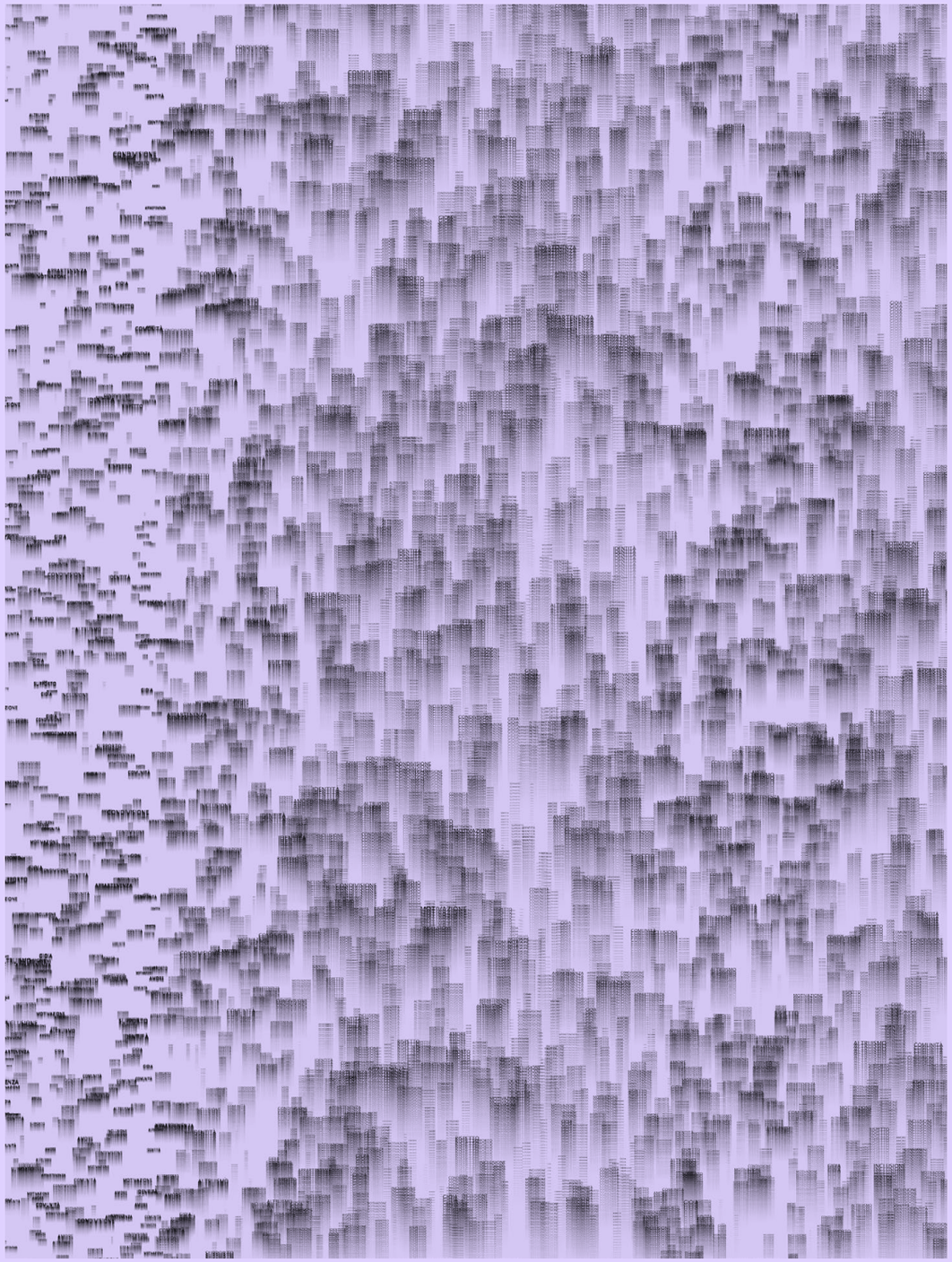
Or Fleisher, Kyle Phillips, and Sehyun Kim for being our mentors.

Stalgia Grigg and Kate Hollenbach for answering our p5.js shader implementation questions.

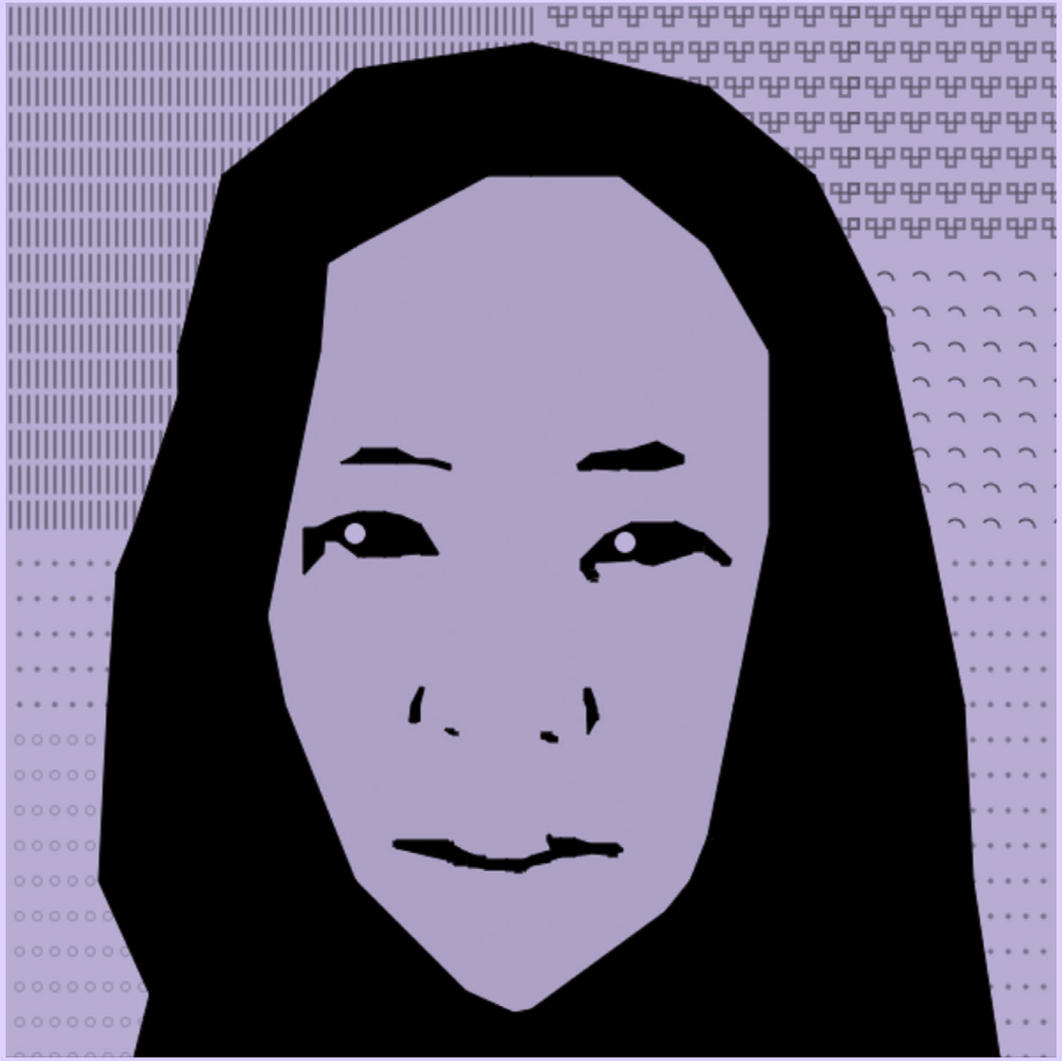
Patricio Gonzalez Vivo and Jen Lowe for writing the amazing Book of Shaders.



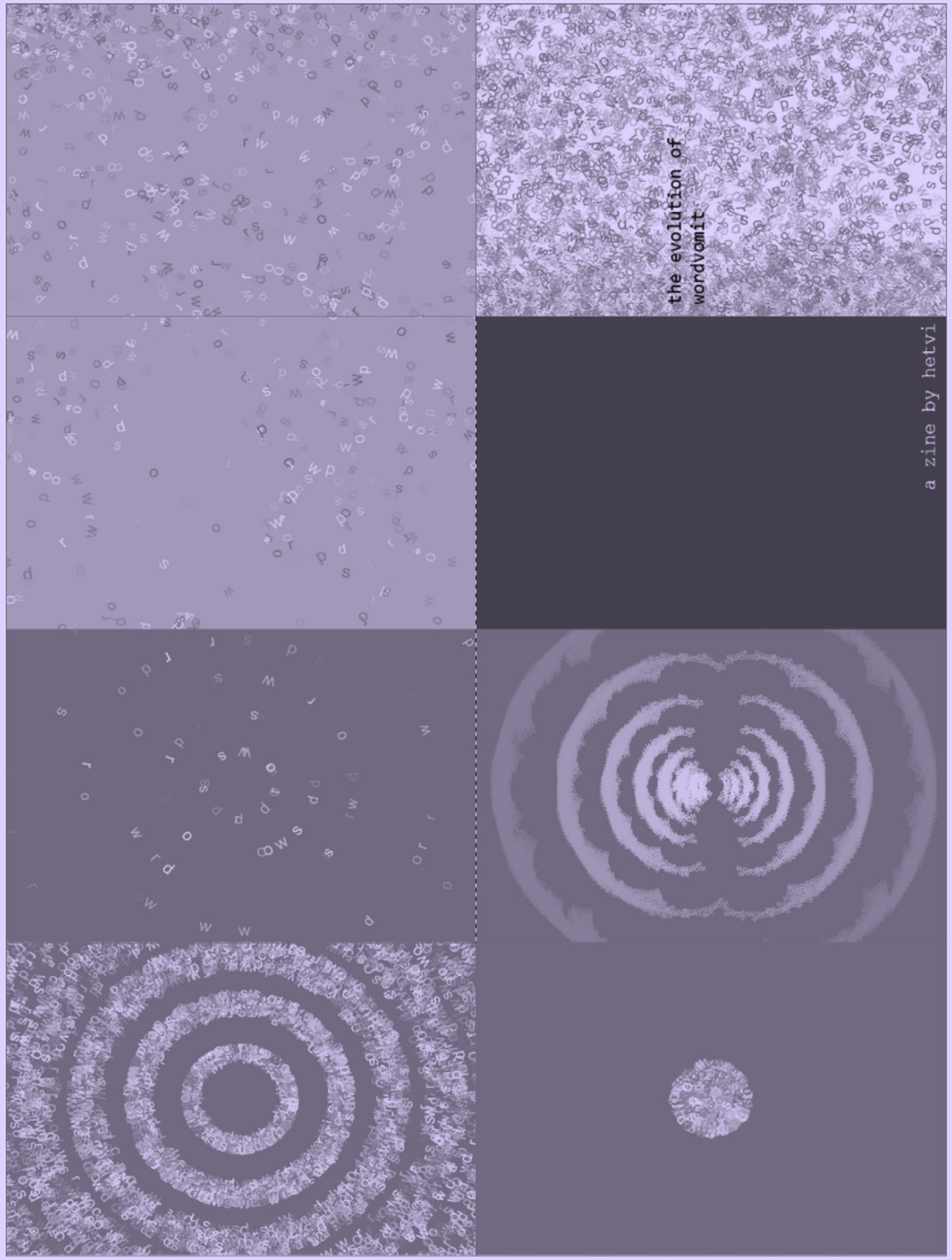








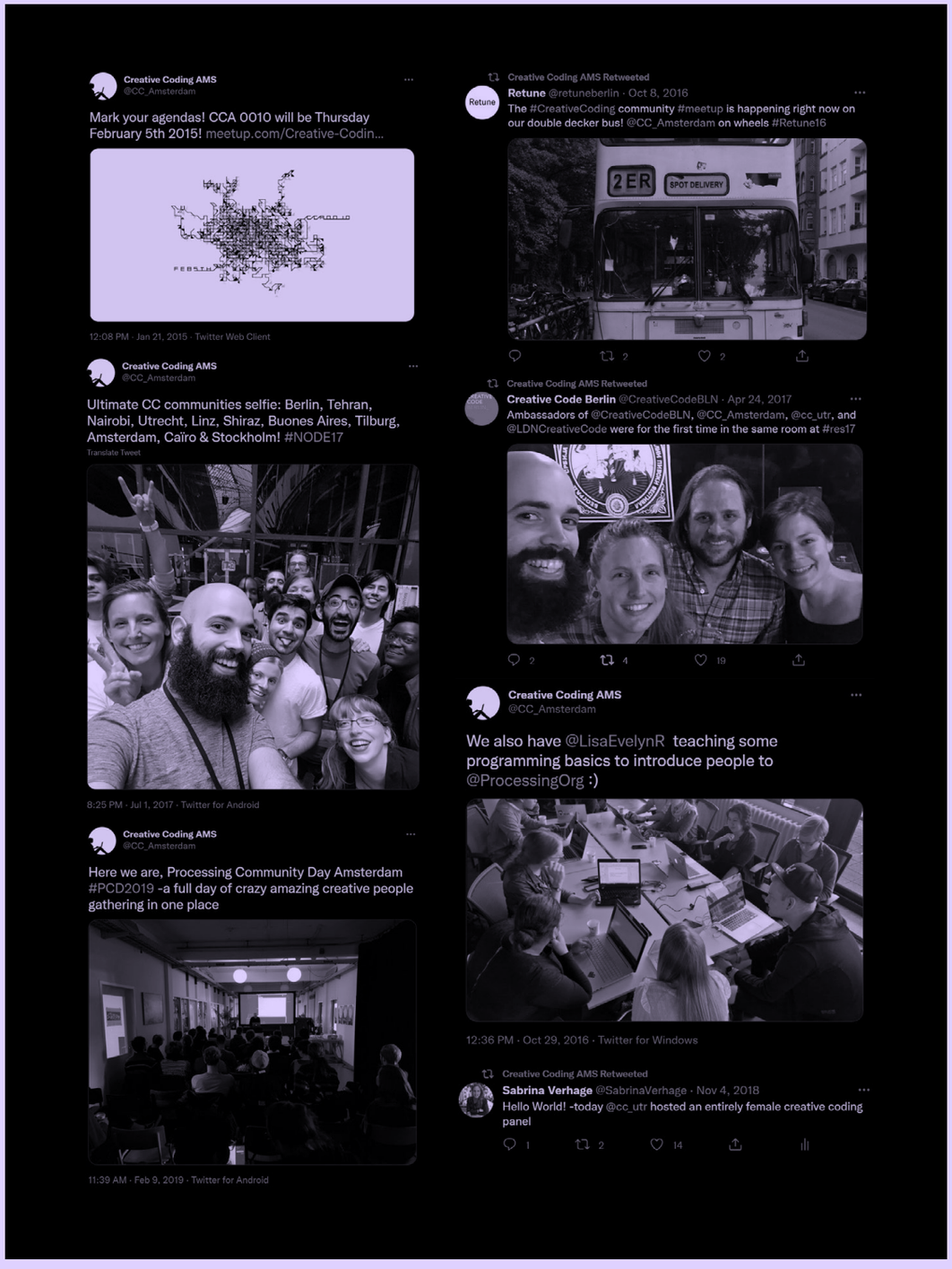



the evolution of
wordvomit
a zine by hetvi


POPRONDE 2016
Open Image
Save Image
Export PDF
3a Mostra LiveCinema
Oi Futuro Ipanema,
Rio de Janeiro
05/12 - 20h
KAUE COSTA // WWW.KAOX.TV
kaox
.tv
THANKS PROCESSING 20 YEARS MADE WITH LOVE!




CREATIVE CODE
STAMMTISCH
19
XDS
MAY 1ST
CREATIVE CODE
STAMMTISCH
20
CO.UP
JUNE 5TH
SCHOOL
MACHINES,
MAKING
&MAKE-BELIEVE







# Creative Coding Madrid_

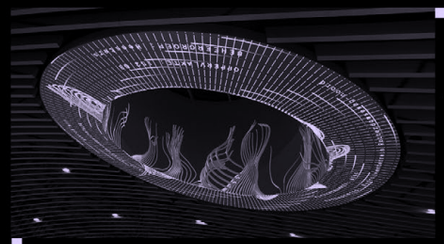
Hola! Somos Creative Coding Madrid y estamos muy contentas de haber llegado hasta aquí formando una comunidad alrededor de la programación creativa que hemos visto crecer. Comenzamos en 2012 bajo el nombre de ProcessingMadrid con la intención de imitar iniciativas similares que surgieron en ciudades de todo el mundo como París, Berlín, Vancouver, El Cairo, etc.

Desde entonces han pasado artistas y programadoras/es de diferentes campos como el videomapping, instalación, arquitectura, audiovisual, robótica o el diseño. Los encuentros que hemos organizado han ido desde talleres y charlas, hasta jams y convocatorias online.

No tenemos ninguna organización detrás que nos patrocine, si bien recibimos apoyo por instituciones de la ciudad, que nos ceden su espacio y recursos para nuestras actividades como Medialab Prado, Matadero Madrid o MakeSpaceMadrid.

Agradecemos a todas las personas que hacen posible Processing, por todos estos años de creatividad sin límites y que sigamos pudiendo disfrutar por muchos años más! Gracias!

# WHAT DOES BEING PART OF THE CCU COMMUNITY MEAN TO YOU AND YOUR CREATIVE CODING PRACTICE?



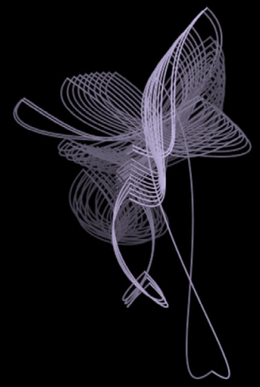
Data-driven projection mapping installation called Stroomtijd, Highlight at Central station Delft

## DEFRAME
(CAROLIEN TEUNISSE)

To meet like-minded people in real-life is not only fun, but it also enriches my network of wonderful and interesting people to collaborate with. It's inspiring to see what other coders make and learn of how they work.

## HAREWOODS
(MATTHIJS HASEBOS)

To my creative coding practice, being part of the CCU community means being inspired.



A piece formed from curves placed by intersecting circles.

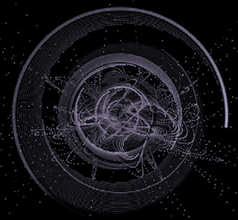
Pendulum (2019), based on #3 by Joost Rekveld at Art Machines 0.2 workshop

## JAKUB VALTAR

CCU is a place to meet fellow creative coders in the Netherlands – they always have some interesting event coming up!

## AUDIOPHOBE
(SIETSE VAN DER MEER)

The CCU community is the bottomless source I turn to for my unquenchable thirst for inspiration, is a trigger to my creativity processes, my main method of extracurricular personal education and above all a warm blanket of social interaction and meeting new friends.



A personal doodle; outtake of a Processing sketch' generating cascades of particle interactions...

CREATIVE CODING UTRECHT

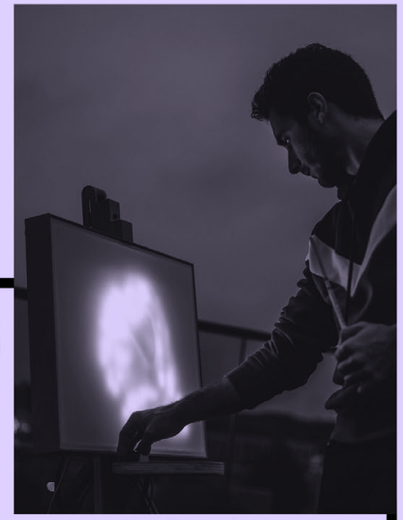


# WHAT DOES BEING PART OF THE CCU COMMUNITY MEAN TO YOU AND YOUR CREATIVE CODING PRACTICE?


The Kinetic Web (2012) is a bookmarklet that turns every website into a mobile sculpture.

## HAY KRANEN

Showing the world that code, algorithms and digital media can be art as well.



Painting with light. Light Canvas, Luis Ferreira

## LUIS FERREIRA

Sharing a passion and inspiring each other.

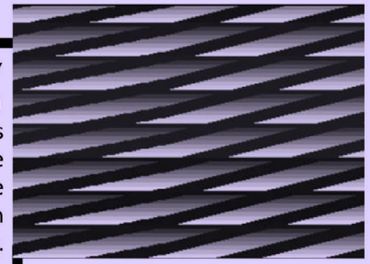
An image of the voidscape 'Zee', it's about chaos and fear.

## KATPATAT
(NEANDER)

CCU brings a network of awesome people and disciplines to the table to collab with and get inspired by.

## F#READY
(FREDDY OFFENGA)

Get inspired by others and interchange creative ideas. Meeting people with the same passion for creative coding.



DSYNC, a sizecoding example in 32 bytes

CREATIVE CODING UTRECHT

# WHAT DOES BEING PART OF THE CCU COMMUNITY MEAN TO YOU AND YOUR CREATIVE CODING PRACTICE?

## NARCODE
(FELIPE IGNACIO NORIEGA)

It means actively learning, applying and contributing to common knowledge and efforts in the digital art sector.

The codeklavier is a system which interprets the piano input as basic lambda calculus functions.

## SASKIA FREEKE

Meeting like minded people, get inspired by each other and support each other.

First time live coding with Timo Hoogland at one of the CCU events

## JO KROESE

I spent my teenage years using Processing mostly by myself. It is beautiful and motivating to be around other people who are pursuing creative coding and pushing their practice.

'Modular Dancing' is a system for creating rule-based modular choreography.

## JONATHAN REUS

A hub for the community of live performers and digital artists to meet and feel a sense of shared space.

Anatomies of Intelligence by Jonathan Reus & Joana Chicau, performance at V2 in 2018

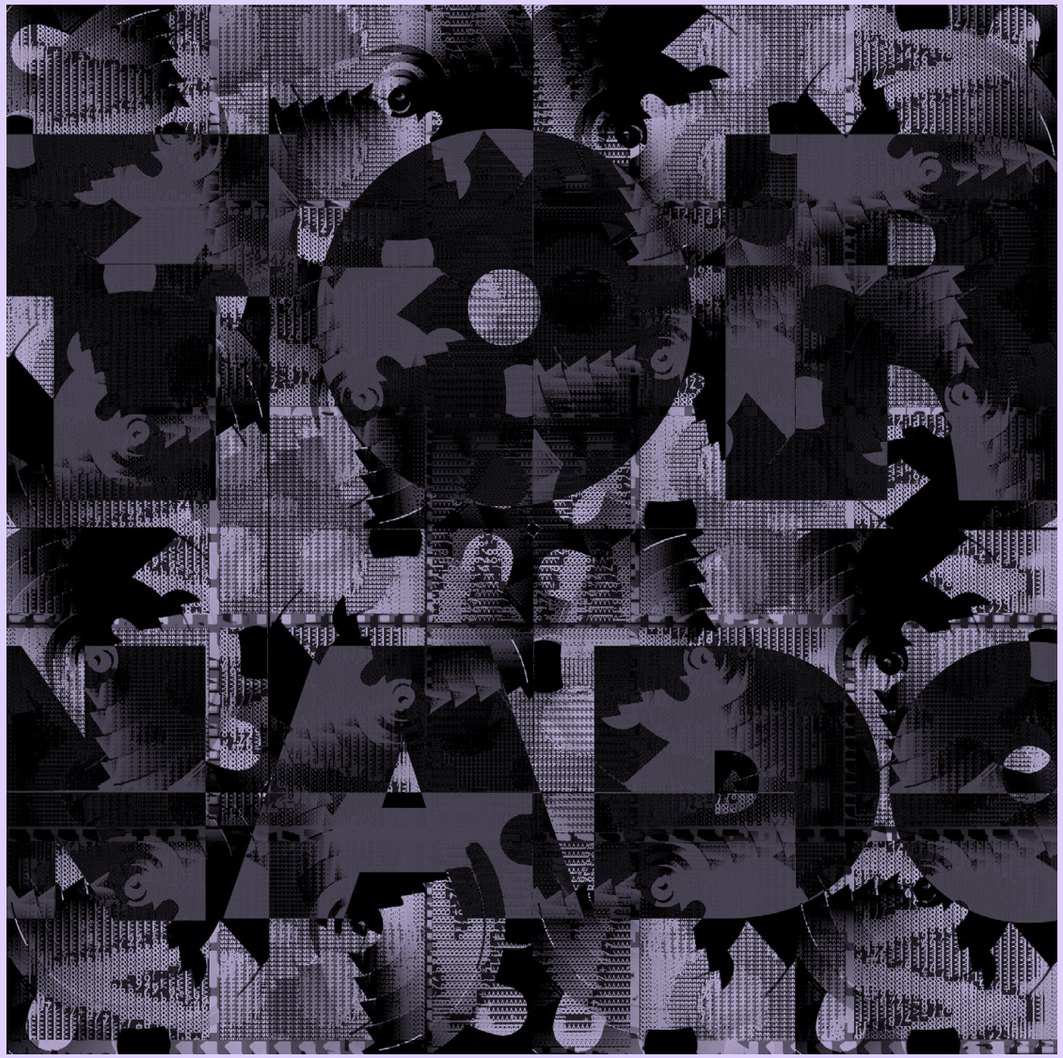
CCU
CREATIVE CODING UTRECHT



```
/* * * * * * * * *
* [a love letter] *
* * * * * * * * * */

let ccc = {

  ada       : "Before I learned to code," +
              "Processing is what I thought coding would be like.",

  samir     : "I've been teaching workshops on it since" +
              "I was a sophomore, like a vast majority of my adult experience" +
              "has been teaching p5. I've seen it since the editor was in alpha.",

  jay       : "Processing was my gateway drug into the world of" +
              "creative coding. Processing opened my mind into" +
              "what is possible with code.",

  garrett   : "p5 took me away from neuroscience and now I just do everything" +
              "on the web. now I do neuroscience on the web, with p5.",

  sarah     : "Processing and p5 are so close to my heart because they make" +
              "coding feel possible and let anyone feel welcome.",

  katherine : "Processing is like your childhood best friend grew up to be" +
              "a celebrity. It's out there doing so many cool things but" +
              "you can still goof around with it.",

}

// yours,
// Creative Code Collective
```


ARCHIVE
2020.01
2020.12
This is our club magazine. The cover and the magazine are all written out using Processing source code. GitHub Repository: ARTiS-CreativeCodingCircle/ARCHIVE-202001-202012
Creative Coding Circle
ARTiS





## Data-Masks : Generator-Discriminator Prototype

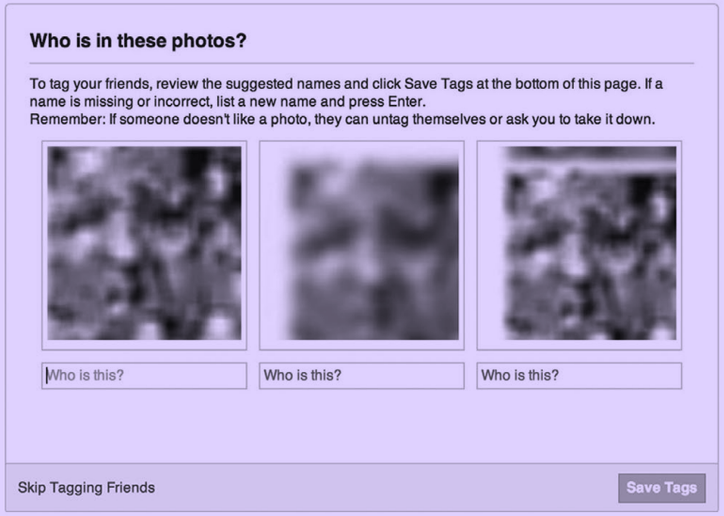
Medium: Processing
Dimensions: 300x400px
Date: March 2013, reformatted in April 2021.

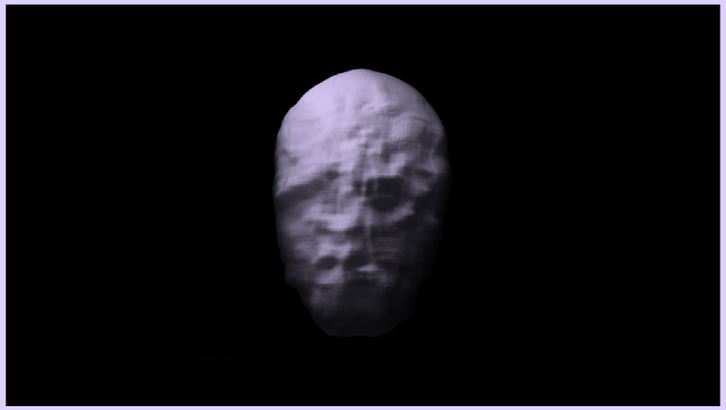
This was the first prototype that eventually inspired me to create my 'Data-Masks' Masters of Science thesis, which uses facial recognition and detection algorithms as its discriminator, rather than a simple linear correlation.

http://www.sterlingcrispin.com/data-masks.html

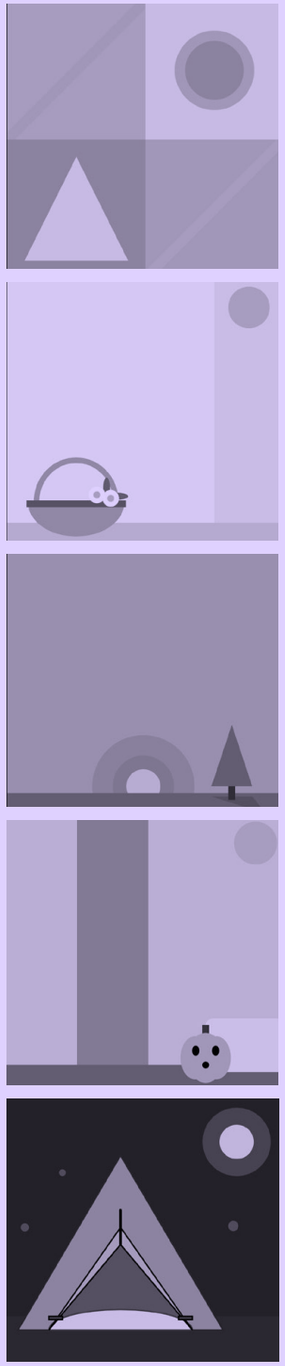
code available here:

http://www.sterlingcrispin.com/datamasks/script.js







NYC Department of Education
COMPUTER SCIENCE FOR ALL

# Creative Web

## A Middle School Curriculum

Developed by the NYC DOE computer science education team, Creative Web is a 54-hour middle school curriculum that teaches students to build computer applications and media that run in a browser. Students will be introduced to HTML and CSS and then learn how to make their own expressive and interactive web pages using the open source JavaScript library p5.js. They will gain a deeper understanding of the relationship between these markup and programming languages, and learn how they can contribute to the larger CS community that exists outside their classrooms.

This course has been implemented in NYC schools, revised by classroom teachers with guidance from the Processing foundation, and aligns with the CS4All Blueprint for CS education that emphasizes a hands-on approach called creative computing. You can view the full curriculum at the link below:

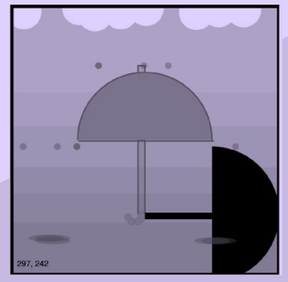
**www.blueprint.cs4all.nyc**

NYC Department of Education

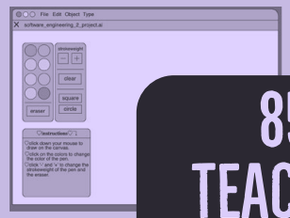
COMPUTER SCIENCE FOR ALL

# INTRODUCTION TO COMPUTATIONAL MEDIA

## A HIGH SCHOOL CURRICULUM




Developed by the NYC DOE computer science (CS) education team, the Introduction to Computational Media is a yearlong (108 hours) creative computing course for high schools using the open-source Javascript library p5.js. By understanding how code can be a medium for creative expression, students will learn the fundamentals of computer science while designing and prototyping interactive projects that run on a browser.



**85+ TEACHERS TRAINED**

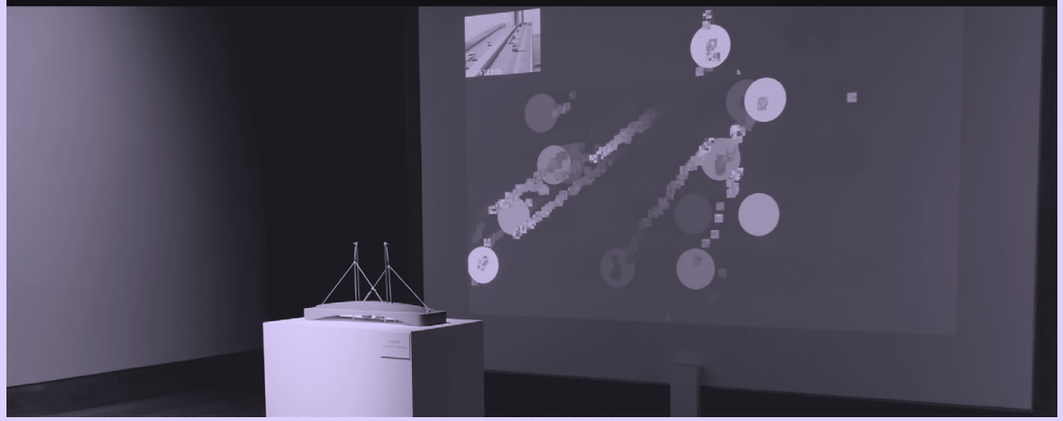

This course has been implemented in NYC schools via CS4All's Software Engineering Program (SEP), revised by classroom teachers with guidance from the Processing Foundation, and aligns with the CS4All Blueprint for CS education that emphasizes a hands-on CS approach called creative computing. You can view the full curriculum at the links below:

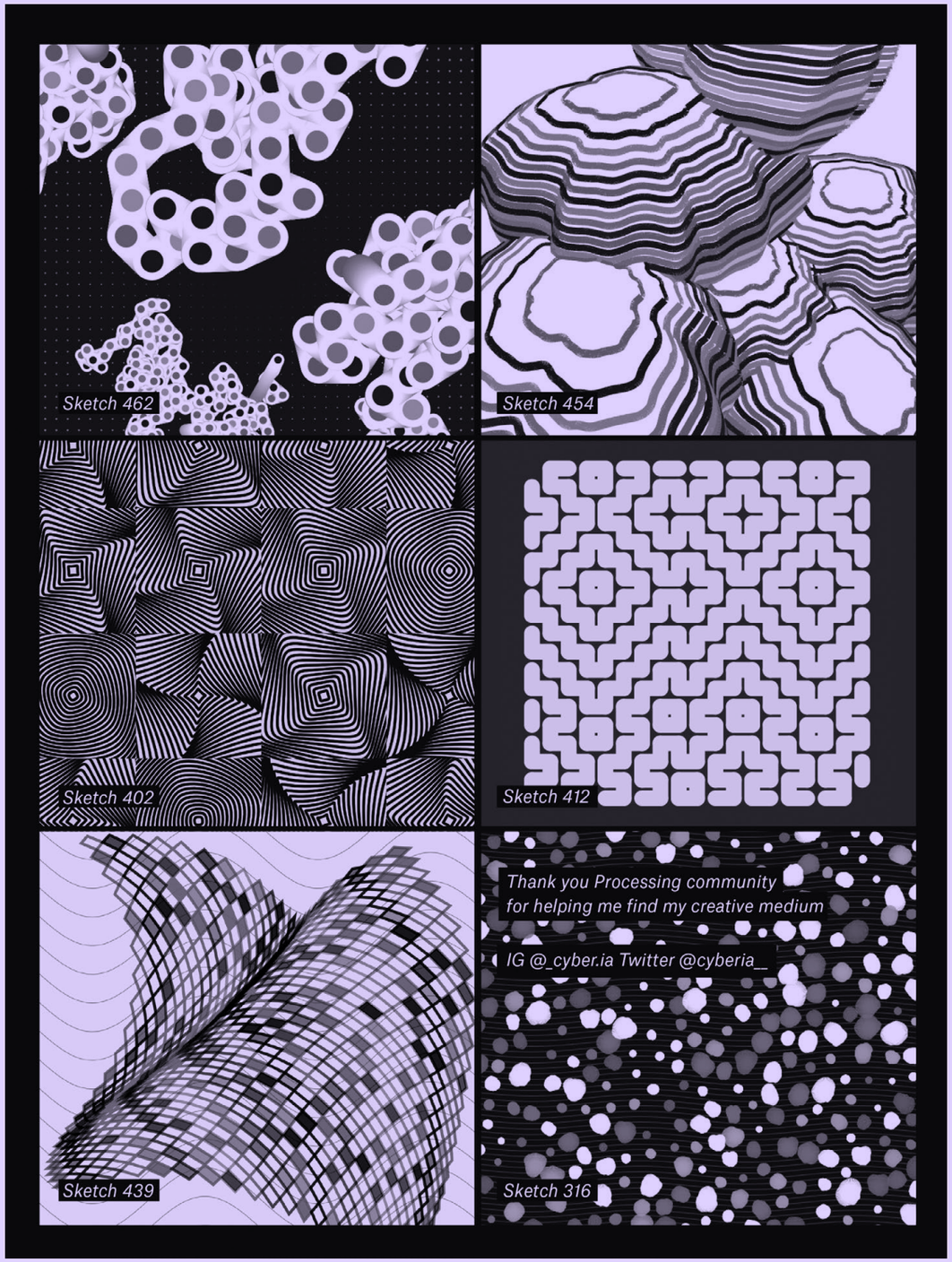
https://tinyurl.com/icmcs4all
https://www.blueprint.cs4all.nyc/




OK! Transmit University of Tampa
PCD 2019

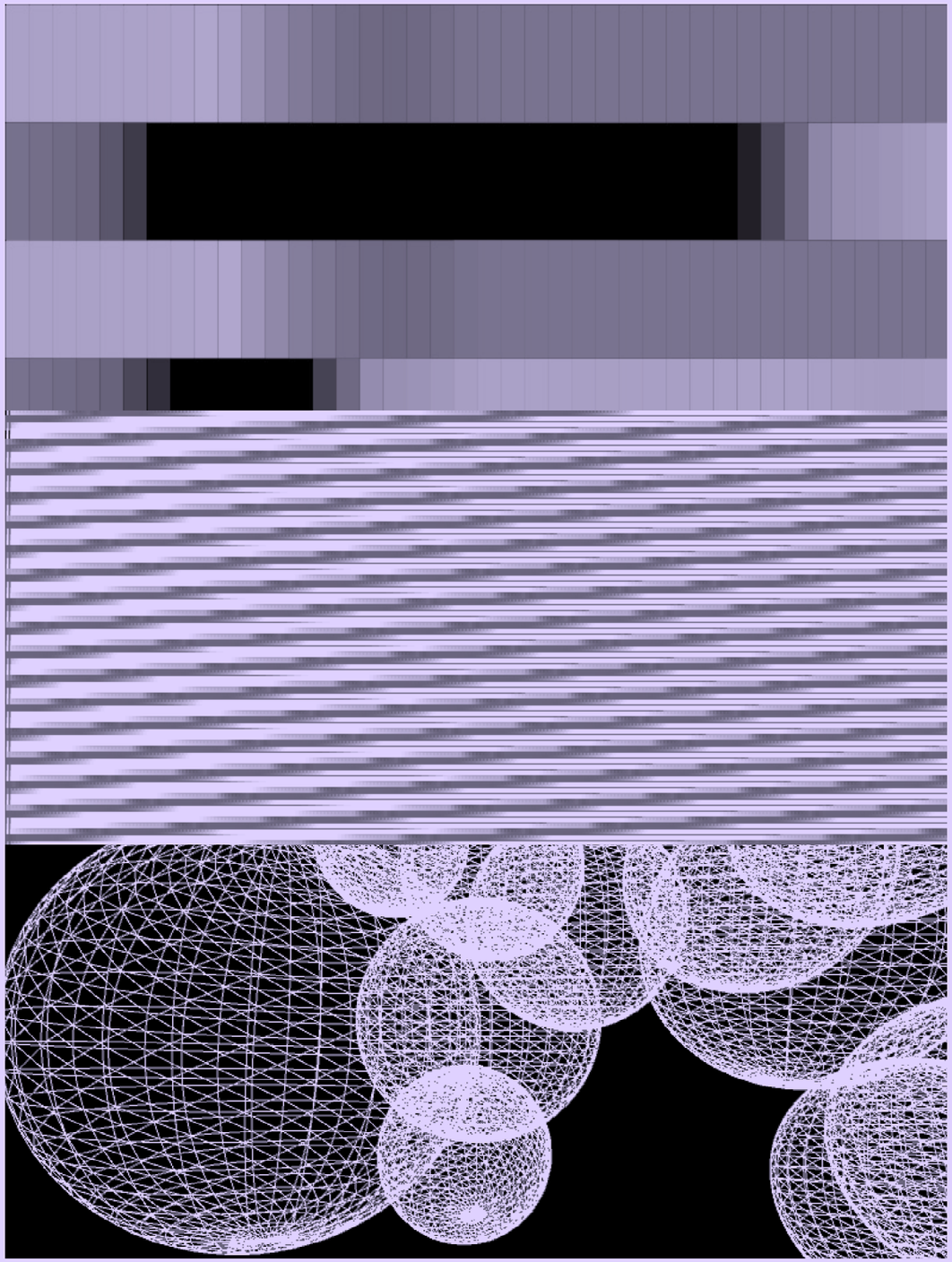
From first exposure to p5.js in 2019 to having a Processing powered generative piece at The Ringling in Sarasota in 2 years. Extremely grateful to the Processing community - especially Mikhail Mansion (pictured, instructor) and Olivia Mansion (@oktransmit) for their leadership and community building.
- James Curran (@minusminusmusic)



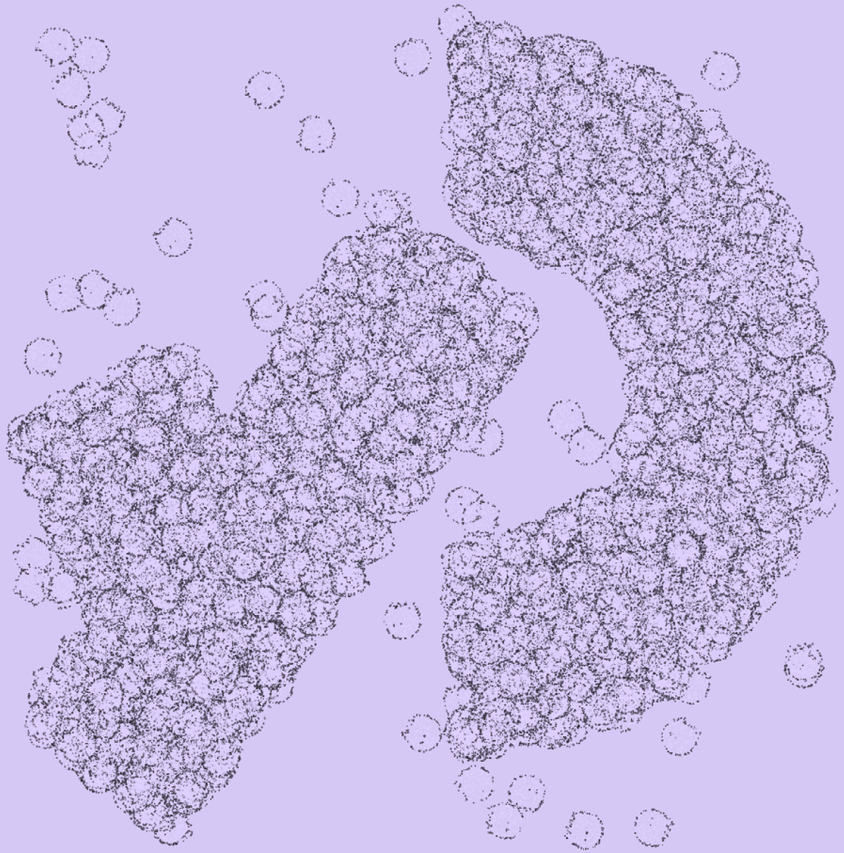

Sketch 462
Sketch 454
Sketch 402
Sketch 412
Sketch 439
Thank you Processing community
for helping me find my creative medium
IG @_cyber.ia Twitter @cyberia_
Sketch 316




Starry Day and Night
Van Gogh Jitterbug
Random art generation
Chandler Zausner 2020





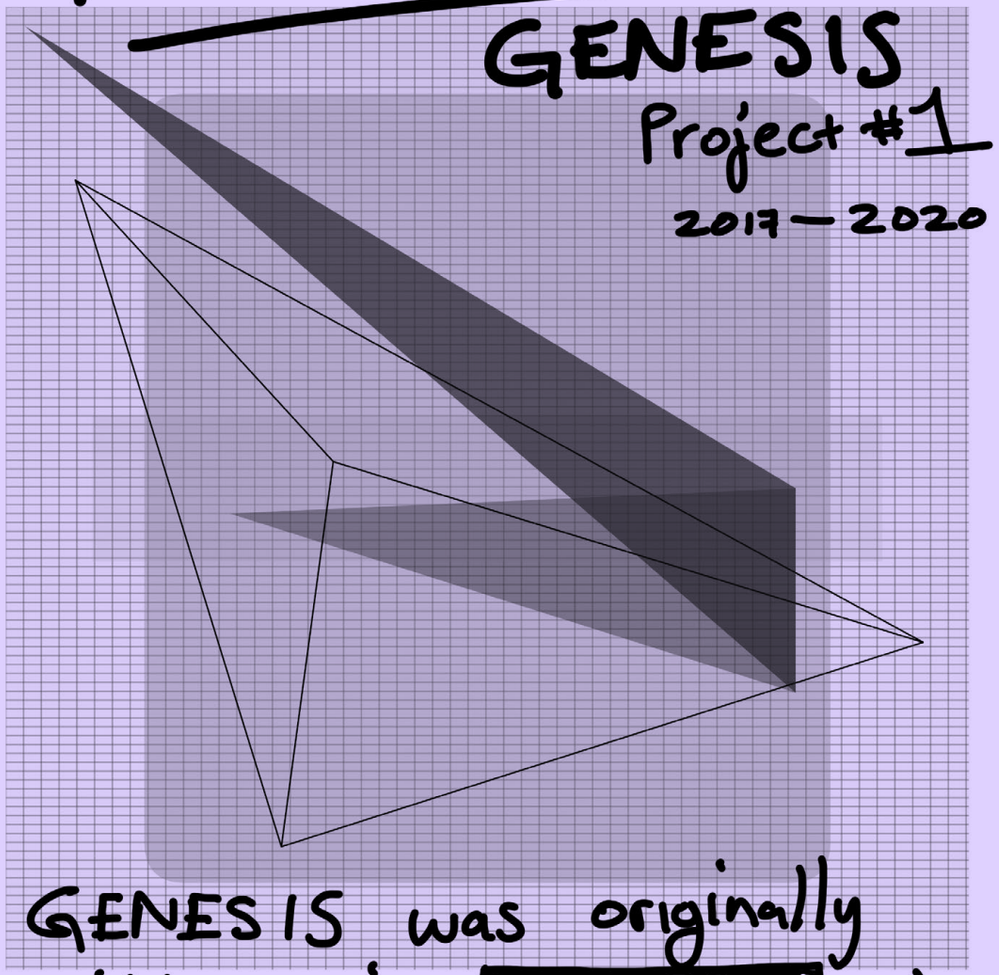


# ART BLOCKS

## GENESIS

Project #1

2017—2020



GENESIS was originally written using **PROCESSING** and was ported to javascript using that good ol' processingjs library before putting on da ETH BLOCKCHAIN



# Journey so far : Processing Fellowship 2019

Manaswini Das Follow
May 12, 2019 · 5 min read

It's been really long since we posted an update of all that has been taking place since March due to some avoidable and unavoidable circumstances. But it's better late than never✌🏻.

I, along with co-fellows Nancy Chauhan and Shaharyar, had the following goals in mind at the beginning of the fellowship:

1. Translation of the p5.js website to Hindi
2. Hindi translation of the p5.js YouTube tutorial series
3. Creating awareness among NGOs and schools about Processing tools in the Indian community

As far as these goals are concerned, we have come a really long way! But this journey wasn't smooth, we faced a lot of ups and downs during the course of this fellowship.

Despite our busy schedule owing to our graduate studentship, Nancy Chauhan and I decided to have a head-start by getting started with the P5js website translation. No wonder, it took us some time to figure our way out. But, thanks to the extremely warm Processing Foundation for extending a helping hand to help us overcome all the challenges that we faced.

lauren mccarthy helped us with the baby steps. Both of us started dividing the work between ourselves and embarked on our very first step towards our mission. We changed the YML files corresponding to individual web pages and committed our changes. In order to make our work cleaner, I squashed the commits and created a pull request. Unfortunately, when I tried to run the project, it crashed.





This was the first blow that we faced. But later, we realized this wasn't the floor of the valley. We tried to figure out the solution to this issue but all of our efforts went in vain. We also had other goals to focus on and this was the first step and now, the first failure as well. Shaharyar came to our rescue and was quick to figure out that the pitfall lied with escape sequences. It hadn't dawned upon us yet🤯. Our translation didn't work due to the improper placement of escape sequences.

At this point, he embarked on a new journey, taking grammar into account, this time. It needs a lot of courage to start afresh! All thanks to him, he worked in that direction and instilled new confidence in us. Cheers to his indomitable will🪨!

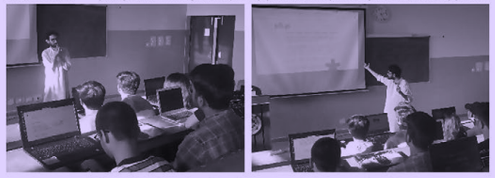
Now, a bulk of the mission was still untouched. We had to reach out to NGOs and schools and we knew that it would require a lot of effort and time to convince them. Nancy had once tried to get started with the teaching stuff and she had to face a lot of questions regarding why students should even be interested for a workshop that involved programming. Our mentor Mathura M Govindarajan was quick to extend a helping hand by connecting us with Aditi Kumar of Feminist Approach to Technology, who was well versed with the nitty gritty of conducting educational workshops for underprivileged kids.

We scheduled a meeting with Aditi and she asked us to craft a proposal with all the stuff that we were going to teach, fairly detailed and gave us a lot of tips for tackling questions that we were likely to face. This is a link to the proposal: https://docs.google.com/document/d/1Y4pN80K8BurujNOnNPPdh0Du_tCTQ2JMKH0s4OMARIk/edit?usp=drivesdk. This helped us a lot in refining and organizing our thought regarding what we wanted to teach.

First, we wanted to start with the audience who were a bit familiar with programming but not with p5.js and have a first-hand idea of the type of response we are likely to receive. Shaharyar conducted a 5-hour workshop in NIT, Hamirpur introducing interactive coding and importance of p5.js. These are a few snaps of the workshop:







Shaharyar Shamshi conducted a workshop on Introduction to creative coding

Following this, Nancy reached out to to several schools in Himachal Pradesh and conducted a 5-hour workshop in a girls senior secondary school, which invited girls from several schools. She had a very lovely experience teaching inquisitive young minds, answering their queries and providing food for their thought. Here are a few snaps of this workshop:

Nancy Chauhan reached out to girls from various schools and conducted a workshop

Conducting a series of such workshops gave us a lot of confidence and direction regarding the tremendous potential these young minds possessed. Now, it was my turn. I conducted a 5-hour workshop for the fresher students of CET, Bhubaneswar who were yet to find their way in programming and highlighted how p5.js could help them start and grow as a developer. These are a few snaps of the workshop:

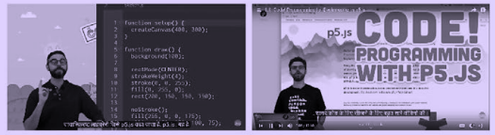




Manaswini Das conducted workshop for the fresher students of CET, Bhubaneswar

Most of the above workshops rooms took place in April. By now, Shaharyar has completed the web translations and it will soon be available in the website. I'm including a link to his contributions.

As far as the YouTube tutorials translation is concerned, Nancy scheduled a meeting with Daniel Shiffman who helped us in getting started with the translation part. Till now we have translated a total of 4 videos and all other videos are to be translated by Nancy and me.

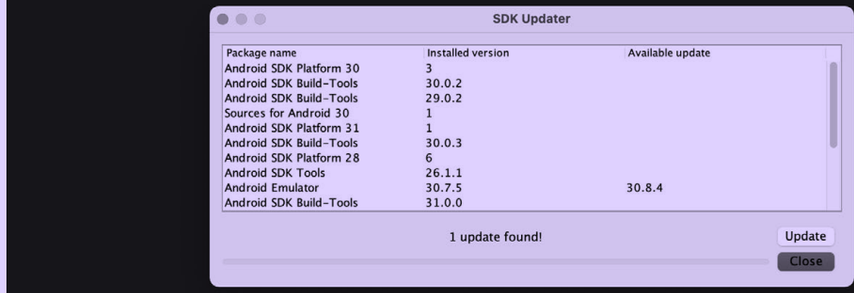



Summing up, a total of 30–40 hours was devoted for web translation collectively, a total of 25 hours for reaching out to various schools, colleges and conducting workshops and a total of 35 hours for YouTube tutorials translation.

We would like to extend our heartfelt gratitude to our mentor Mathura M Govindarajan for her invaluable guidance and support all throughout these 2 months. She made sure that we were on the right track and walked the extra mile to make sure that we received timely help every time we needed. Last but not the least, we would








**Initial commit for SDK Updater** Browse files

master (#359)
latest ··· android-264

**rupak0577** committed on 23 Jul 2017 1 parent 1b9f9ba commit 9dfff6e645718b69d879d81b4b4052365c6389cd

Showing 3 changed files with **354 additions** and **8 deletions**. Unified Split

7 src/processing/mode/android/AndroidEditor.java

4 src/processing/mode/android/AndroidSDK.java

351 src/processing/mode/android/SDKUpdater.java

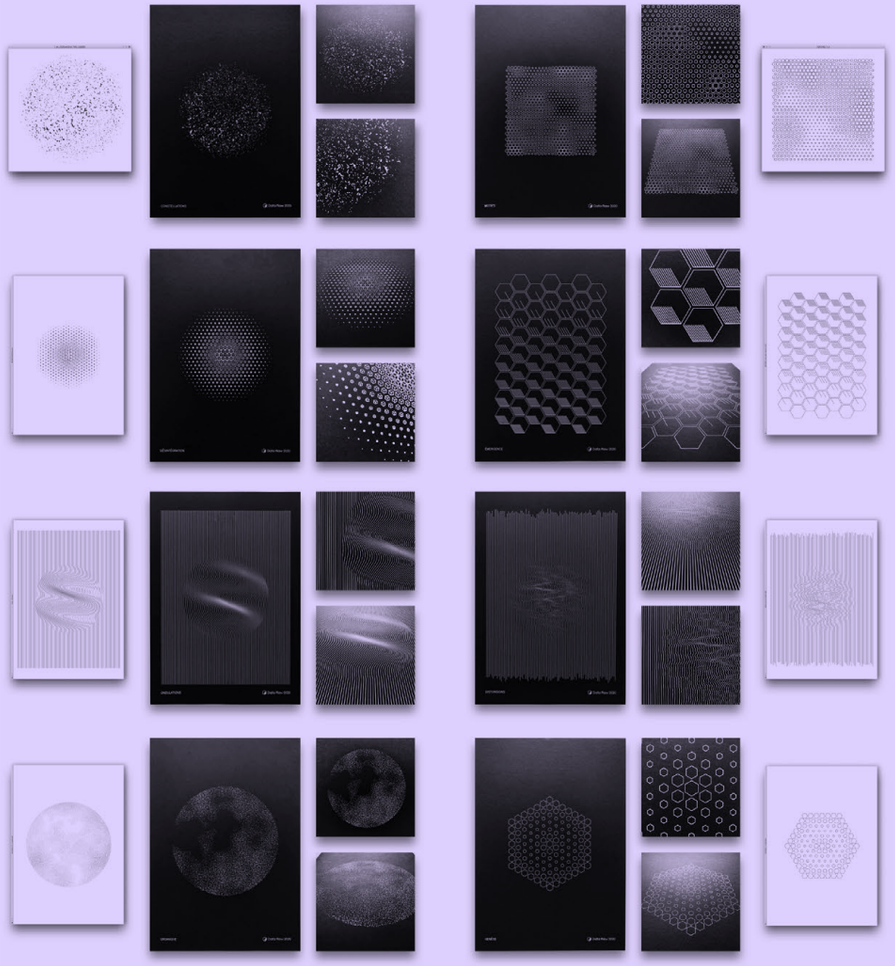
```
@@ -0,0 +1,351 @@
+ package processing.mode.android;
+
+ import processing.app.Platform;
+
+ import javax.swing.*;
+ import javax.swing.border.EmptyBorder;
+ import javax.swing.table.DefaultTableModel;
+ import java.awt.*;
+ import java.awt.event.ActionEvent;
+ import java.awt.event.ActionListener;
+ import java.beans.PropertyChangeEvent;
+ import java.beans.PropertyChangeListener;
+ import java.io.BufferedReader;
+ import java.io.File;
+ import java.io.IOException;
+ import java.io.InputStreamReader;
+ import java.util.ArrayList;
+ import java.util.Arrays;
+ import java.util.HashMap;
+ import java.util.Vector;
+
+ public class SDKUpdater extends JFrame implements PropertyChangeListener {
+
+     private HashMap<String, Boolean> requiredPackages;
+
+     private final Vector<String> columnsInstalled = new Vector<>(Arrays.asList("Path", "Version",
```

# GENERATIVE ART : « TIPS & TRICKS »

## Romain ASTOURIC / @DataFlaw

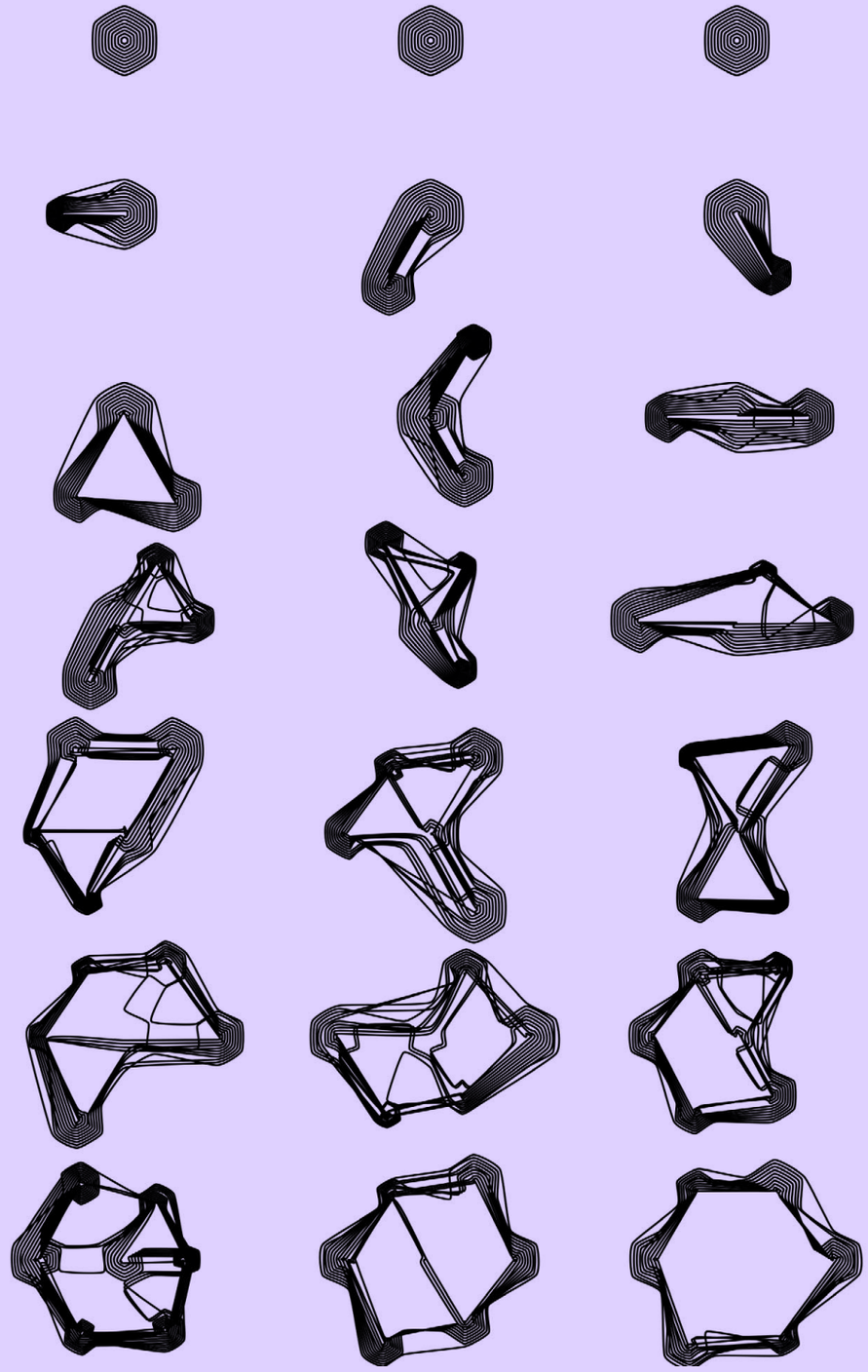
- Generative art series produced with Processing, drawn by an Axidraw A3 / V3 digital plotter -
*White gel ink pen, deep black A3 paper, high density (300g / m2).*

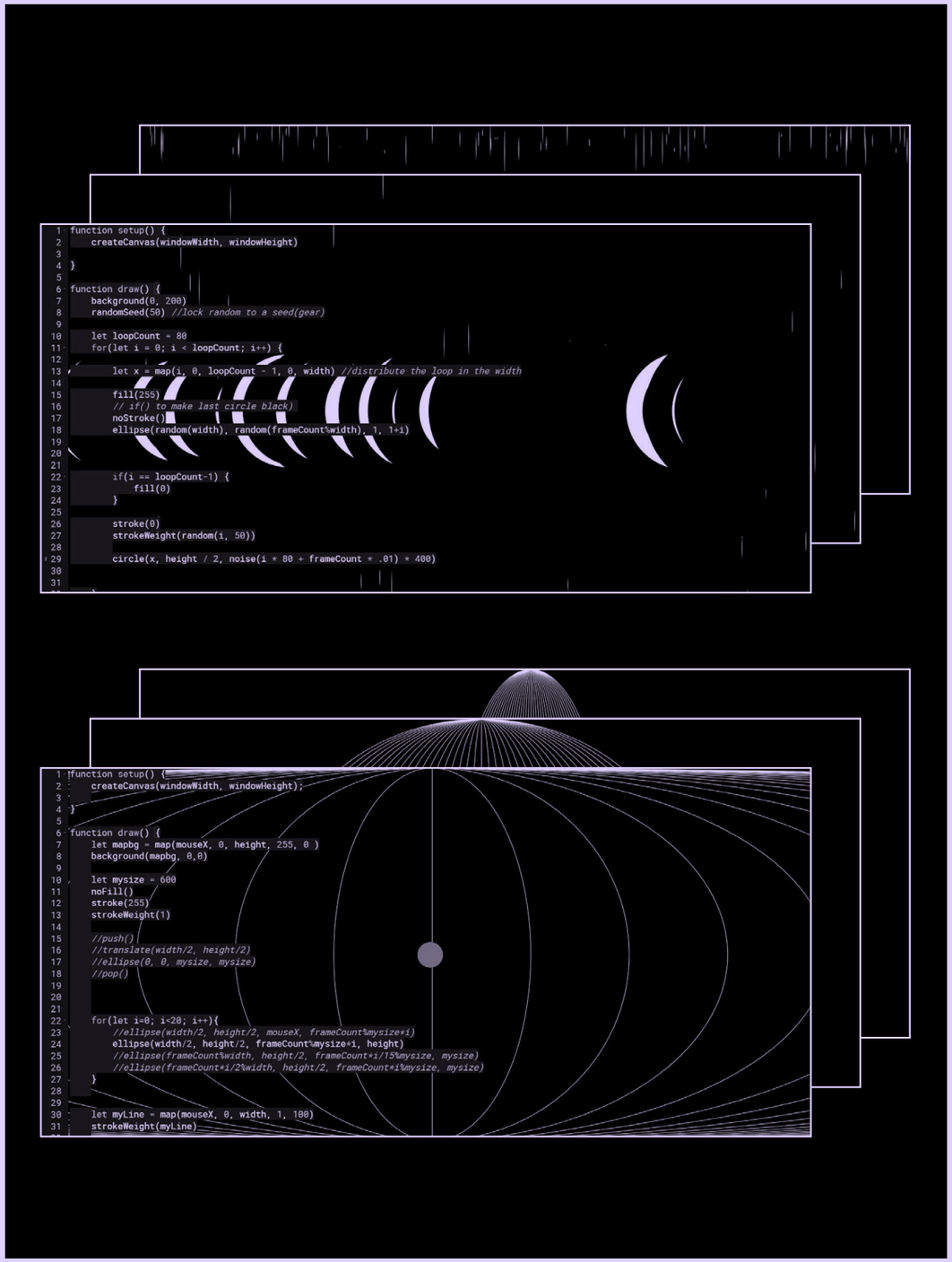
Thrilled by the renaissance of digital plotters, I designed and produced this series of fully coded artworks on Processing. Inspired by classic geometric patterns and various creative algorithms with simple rules but fascinating results, I wanted to show the hidden beauty of complex formulas and abstract calculations.



***dataflaw.art - liinks.co/dataflaw***







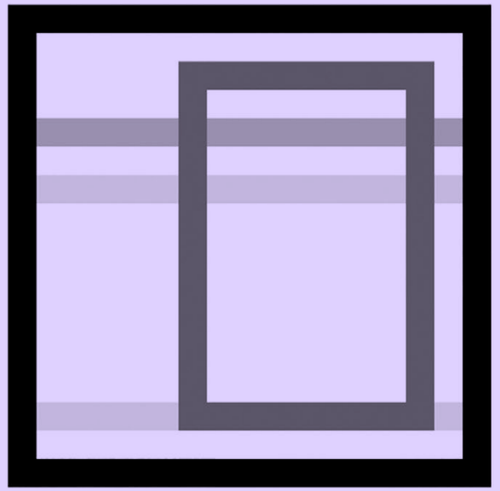
```
function setup() {
    createCanvas(windowWidth, windowHeight)

}

function draw() {
    background(0, 200)
    randomSeed(50) //lock random to a seed(gear)

    let loopCount = 80
    for(let i = 0; i < loopCount; i++) {

        let x = map(i, 0, loopCount - 1, 0, width) //distribute the loop in the width

        fill(255)
        // if() to make last circle black)
        noStroke()
        ellipse(random(width), random(frameCount%width), 1, 1+i)


        if(i == loopCount-1) {
            fill(0)
        }

        stroke(0)
        strokeWeight(random(i, 50))

        circle(x, height / 2, noise(i * 80 + frameCount * .01) * 400)

    }
```

```
function setup() {
    createCanvas(windowWidth, windowHeight);
}

function draw() {
    let mapbg = map(mouseX, 0, height, 255, 0 )
    background(mapbg, 0,0)

    let mysize = 600
    noFill()
    stroke(255)
    strokeWeight(1)

    //push()
    //translate(width/2, height/2)
    //ellipse(0, 0, mysize, mysize)
    //pop()


    for(let i=0; i<20; i++){
        //ellipse(width/2, height/2, mouseX, frameCount%mysize*i)
        ellipse(width/2, height/2, frameCount%mysize*i, height)
        //ellipse(frameCount%width, height/2, frameCount*i/15%mysize, mysize)
        //ellipse(frameCount*i/2%width, height/2, frameCount*i%mysize, mysize)
    }

    let myLine = map(mouseX, 0, width, 1, 100)
    strokeWeight(myLine)
```



# JEFF DAVIS



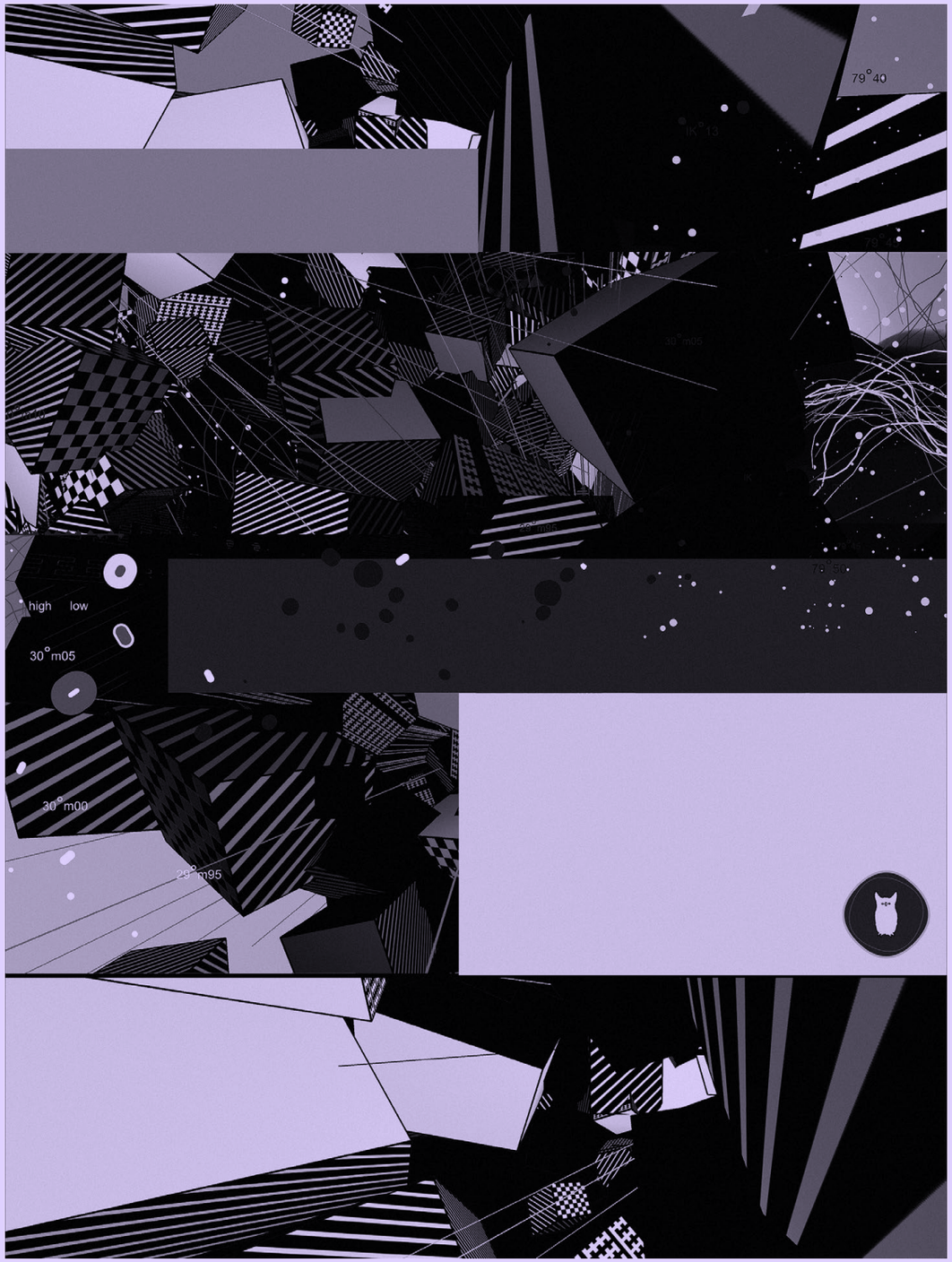


portfolio 1-1

2013, digital c-print
(single edition), 18 x 18 in.

PREV / NEXT

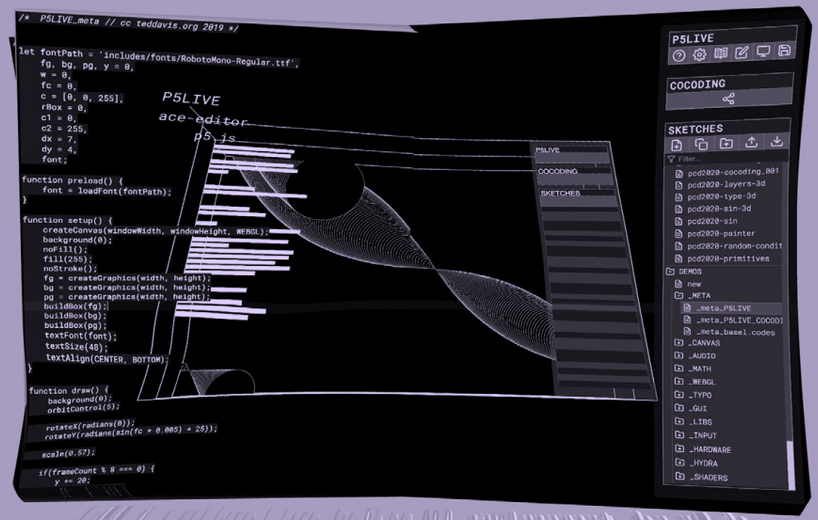
SHOW THUMBNAILS





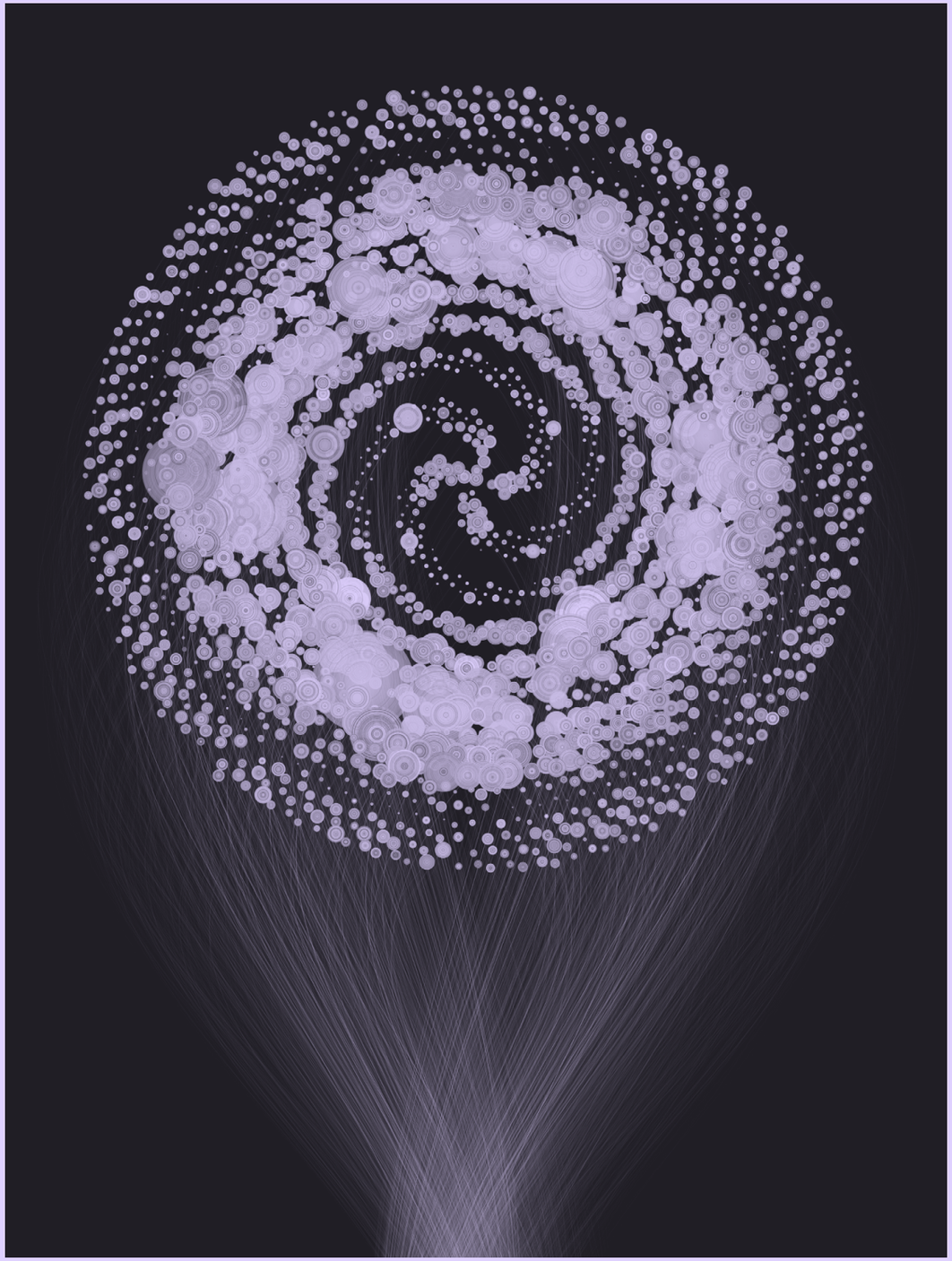


Processing library for vector graphics on vector displays




Collaborative Live Coding VJ environment for p5.js

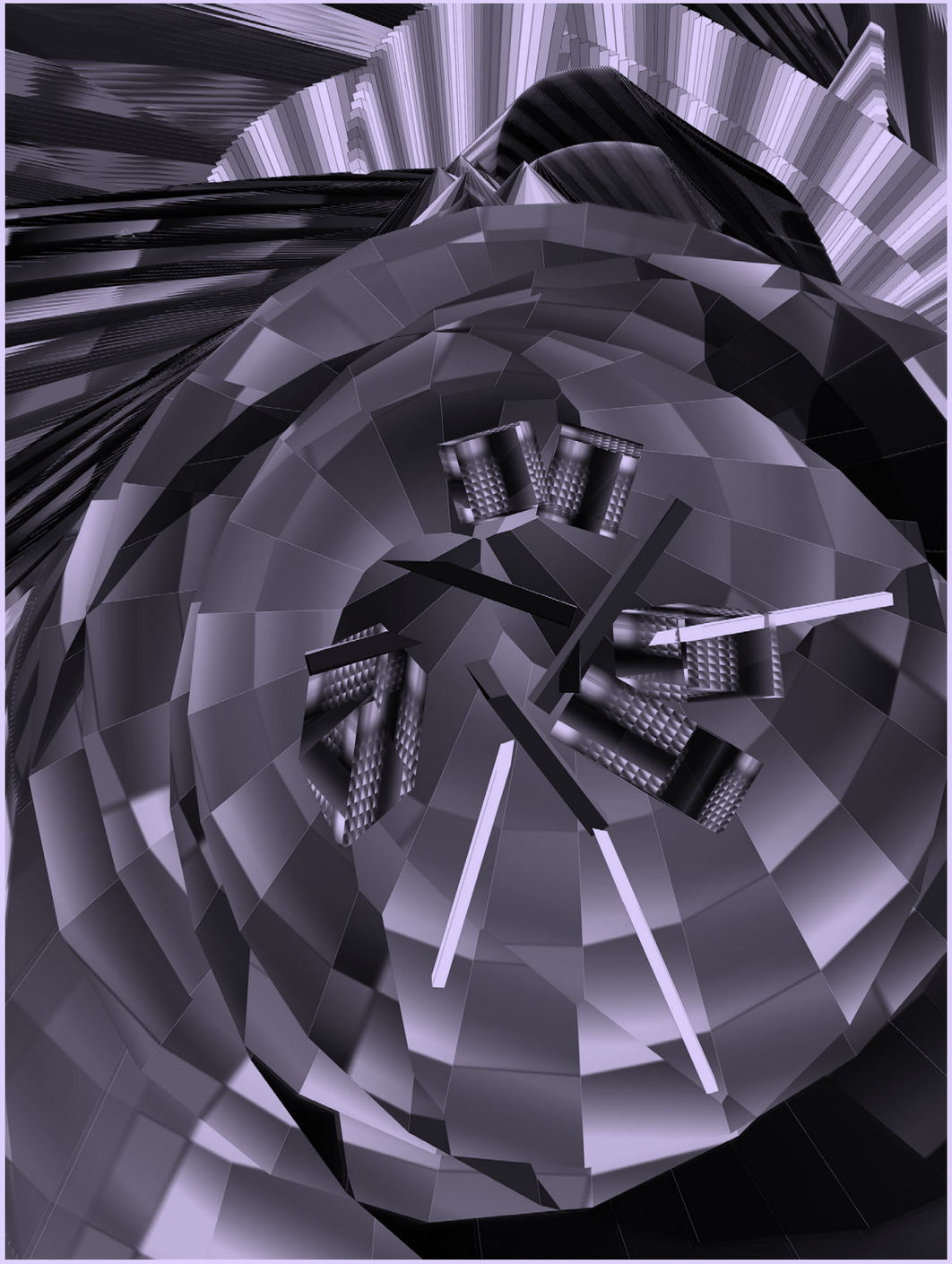

Byte level glitching library for p5js



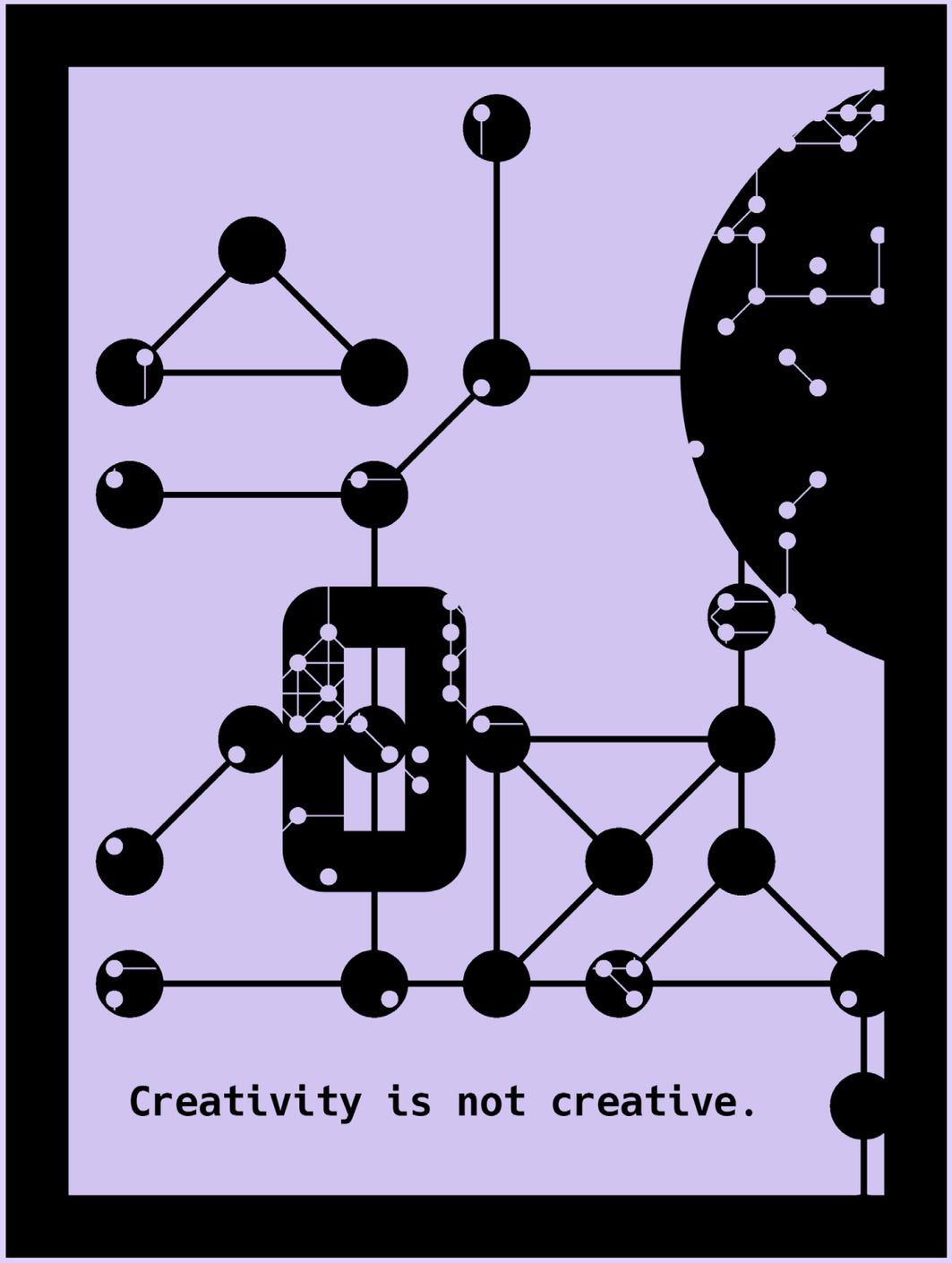


# How Proc essing ch anged eve rything - I started t eaching soft ware to desig n students in 2000. I had jus t moved to Sweden to work as a unive rsity teacher after a fairly short period in t he industry. Back then, Java w as the most used language in academi a, but there was little done in the field of le arning programming for people with no initial interest in engineeri ng or natural sciences. The tools I got access to were clunky, and so was the exp erience students had. Coming from an engineering background myself, it didn't occ ur to me in the first place how much the context of use meant for having a good l earning process. The first year I made the same mistakes my teachers made with us some years prior: Taught theory before doing any practical experiments, not show ing the value of using software in a creative way until having been exposed to th e theoretical background behind the tools. I had nothing to rely on, there were n o other experiences I could find, and very few of my colleagues had any experienc e in using technology as a material and not just as a tool. Building a curriculum was a matter of trial and error encompassing shifting between tools, books, and different versions of Java. And then, a former student taking his masters abroad spoke to me about Processing in late 2004. I had no idea where it came from, I ju st knew that students could download it on their computers and run it in a matter of minutes. The examples covered relevant -yet engaging- topics. Code looked cle an and the API was very straightforward. I was game in a matter of minutes. We co uld use all of the knowledge we had created over the course of years and integrat e it in a much easier to use IDE. Soon enough, I got involved in the creation of Arduino, a project that got influences -among others- from Processing. We used ou r own editor to write, compile, and upload code, but there was a disconnect betwe en tools. If people learned how to program on Processing (by then it was already a de-facto standard at many design and art schools), why should they confront a d ifferent set of metaphors to program microcontroller boards? By the time Processi ng's codebase became free software, and just like others did before, we made Proc essing's UI the point of departure of our own UI. And that is how Processing chan ged everything ... or at least, everything that I do.

How Processing changed everything - I started teaching software to design students in 2000. I had just moved to Sweden to work as a university teacher after a fairly short period in the industry. Back then, Java was the most used language in academia, but there was little done in the field of learning programming for people with no initial interest in engineering or natural sciences. The tools I got access to were clunky, and so was the experience students had. Coming from an engineering background myself, it didn't occur to me in the first place how much the context of use meant for having a good learning process. The first year I made the same mistakes my teachers made with us some years prior: Taught theory before doing any practical experiments, not showing the value of using software in a creative way until having been exposed to the theoretical background behind the tools. I had nothing to rely on, there were no other experiences I could find, and very few of my colleagues had any experience in using technology as a material and not just as a tool. Building a curriculum was a matter of trial and error encompassing shifting between tools, books, and different versions of Java. And then, a former student taking his masters abroad spoke to me about Processing in late 2004. I had no idea where it came from, I just knew that students could download it on their computers and run it in a matter of minutes. The examples covered relevant -yet engaging- topics. Code looked clean and the API was very straightforward. I was game in a matter of minutes. We could use all of the knowledge we had created over the course of years and integrate it in a much easier to use IDE. Soon enough, I got involved in the creation of Arduino, a project that got influences -among others- from Processing. We used our own editor to write, compile, and upload code, but there was a disconnect between tools. If people learned how to program on Processing (by then it was already a de-facto standard at many design and art schools), why should they confront a different set of metaphors to program microcontroller boards? By the time Processing's codebase became free software, and just like others did before, we made Processing's UI the point of departure of our own UI. And that is how Processing changed everything ... or at least, everything that I do.



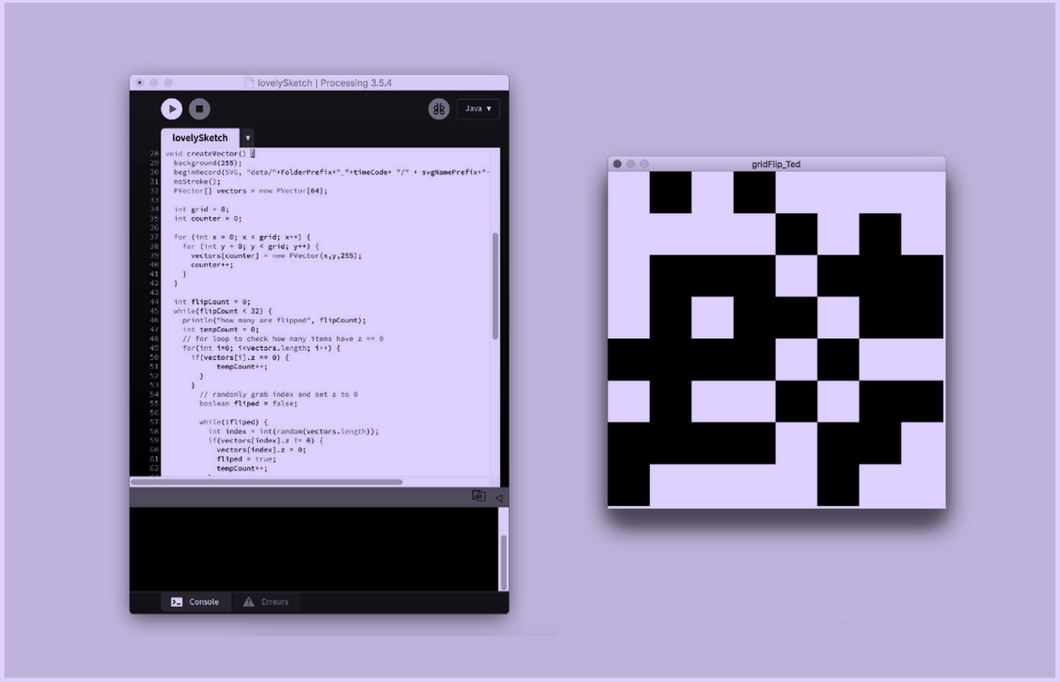


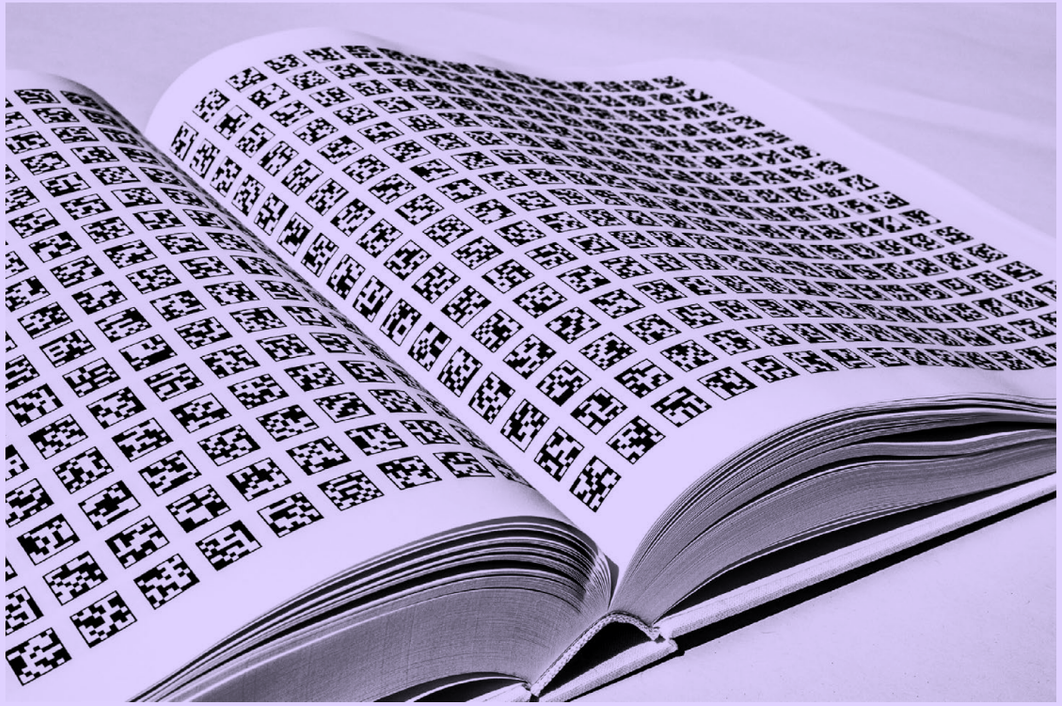

Creativity is not creative.


EPSON


a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p
q
r
s
t
u
v
w
x
y
z




lovelySketch | Processing 3.5.4
Java
lovelySketch
Console
Erreurs
gridFlip_Ted




Fabian Dennler
Developer
Teacher
Coach

All beginnings are hard!
In a programming language there are many confusing elements for beginners: variables, conditions, functions, parameters and more.

In order to illustrate the subject of software development to the students in the vocational training as application developer and thus make it understandable, Processing has proven itself with the creation of simple shapes and their manipulation in terms of size, position and color.

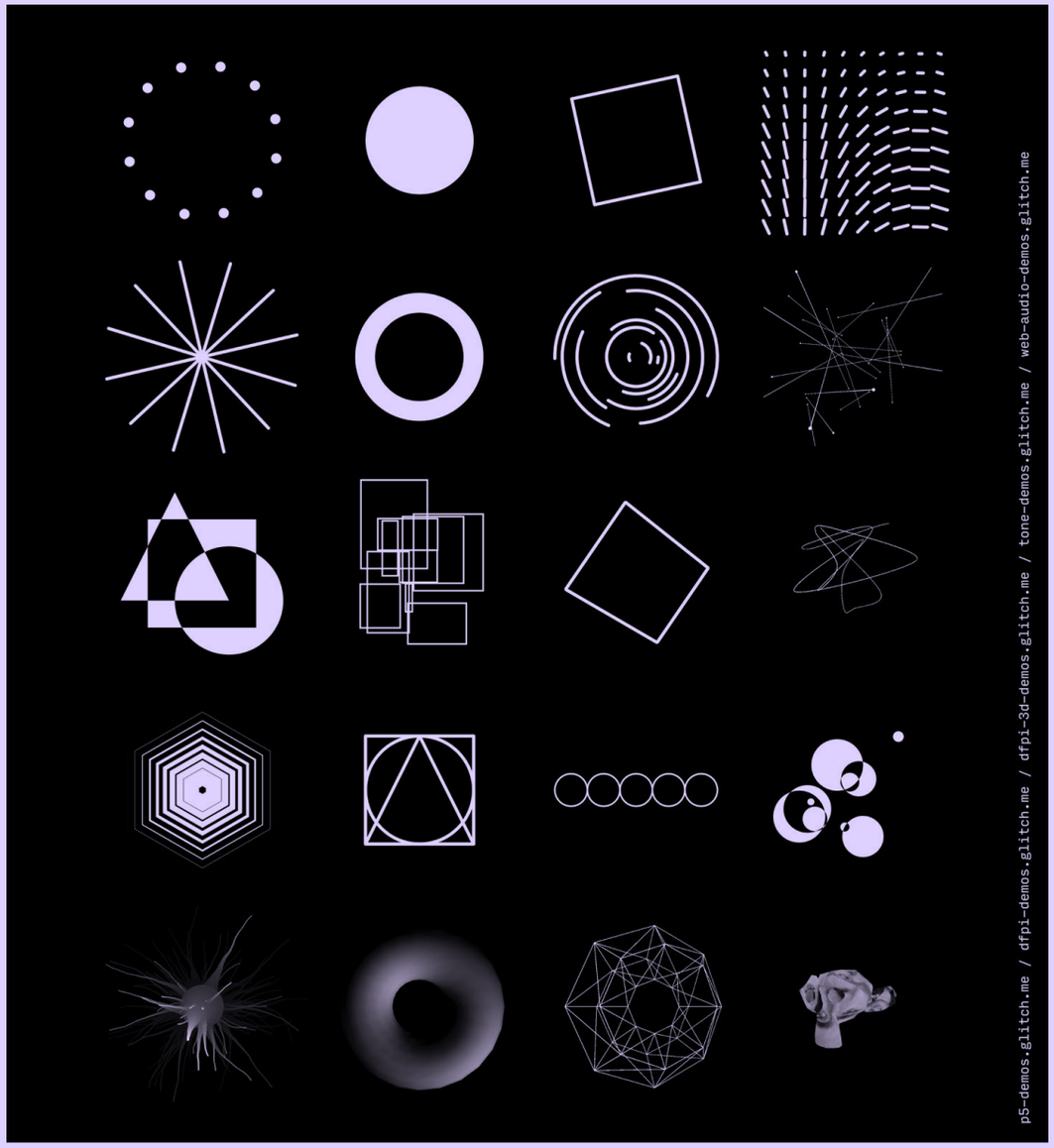
The key to understanding programming lies in learning how to formulate program instructions using the right syntax and how to structure program sequences, i.e. how to design algorithms.

Open tasks favor individual learning and processing paths, promote subject-specific and interdisciplinary competencies and enable discovery and self-directed work and learning.

Processing provides a brain-friendly introduction to the topic. Because it builds on prior knowledge from geometry and mathematics as well as produces a pictorial result, both sides of the brain are activated.

20 years of Processing!
One of my success factors as a teacher and coach, since 2001.

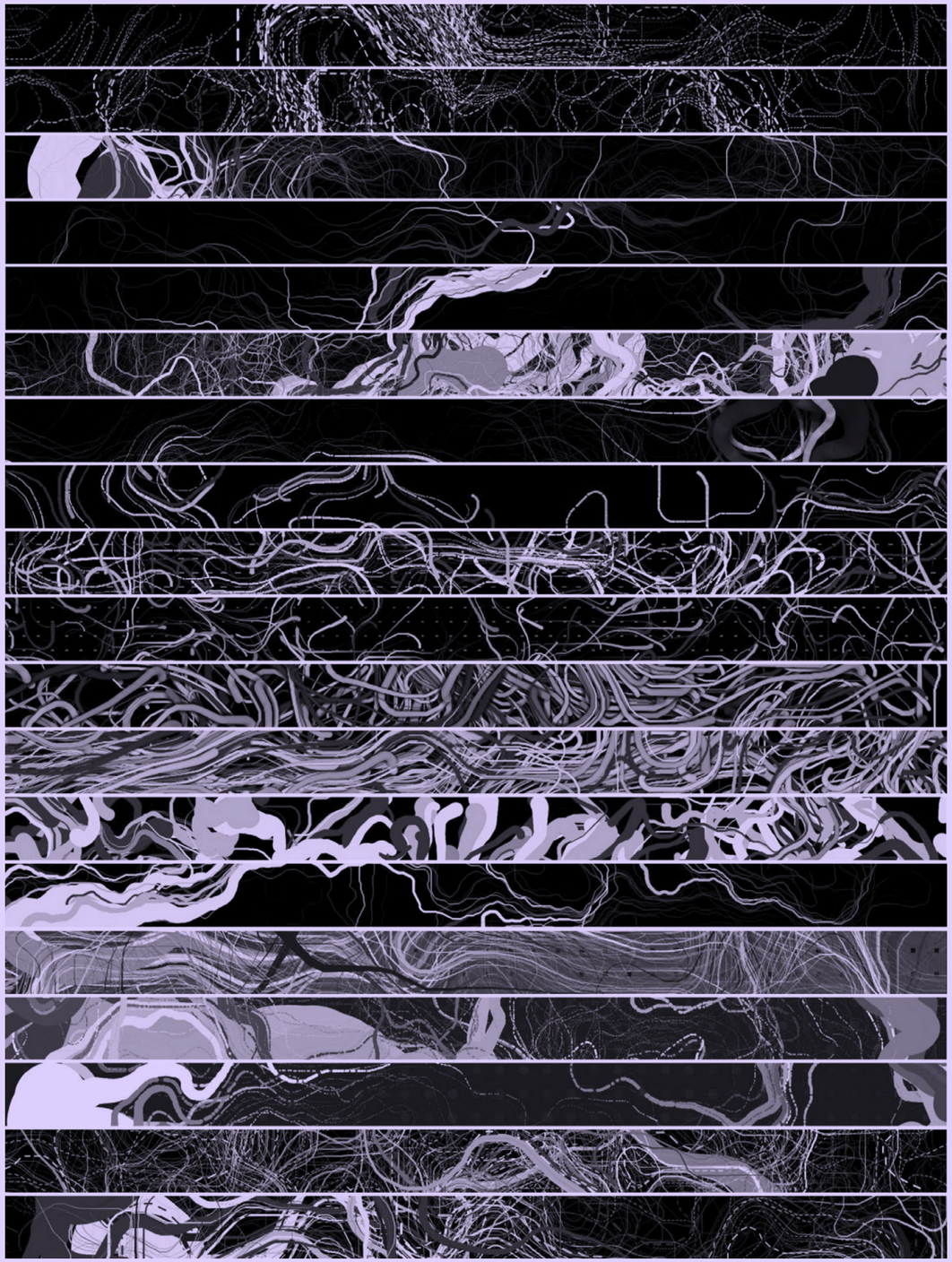




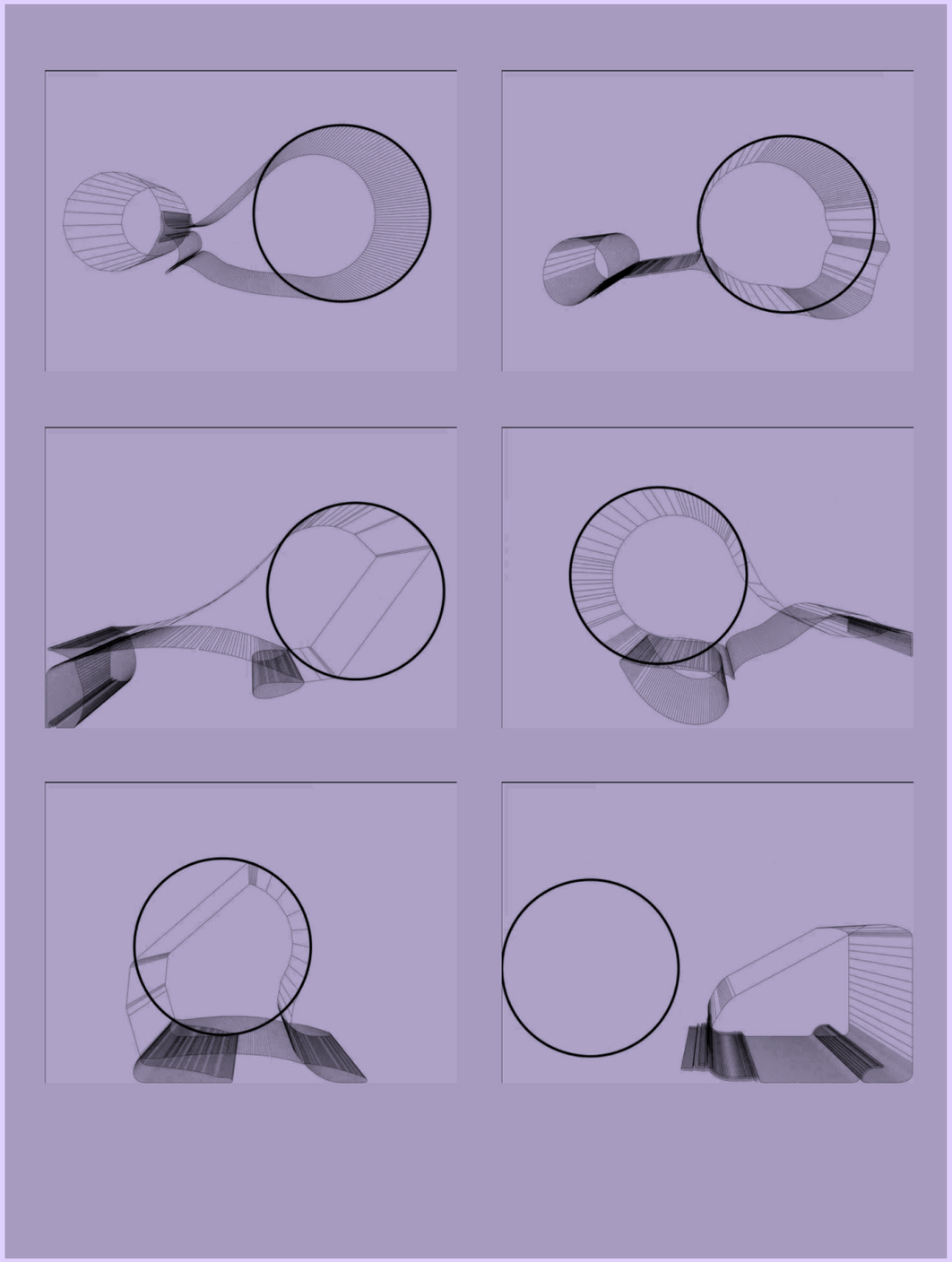

## Happy 20th birthday, Processing!

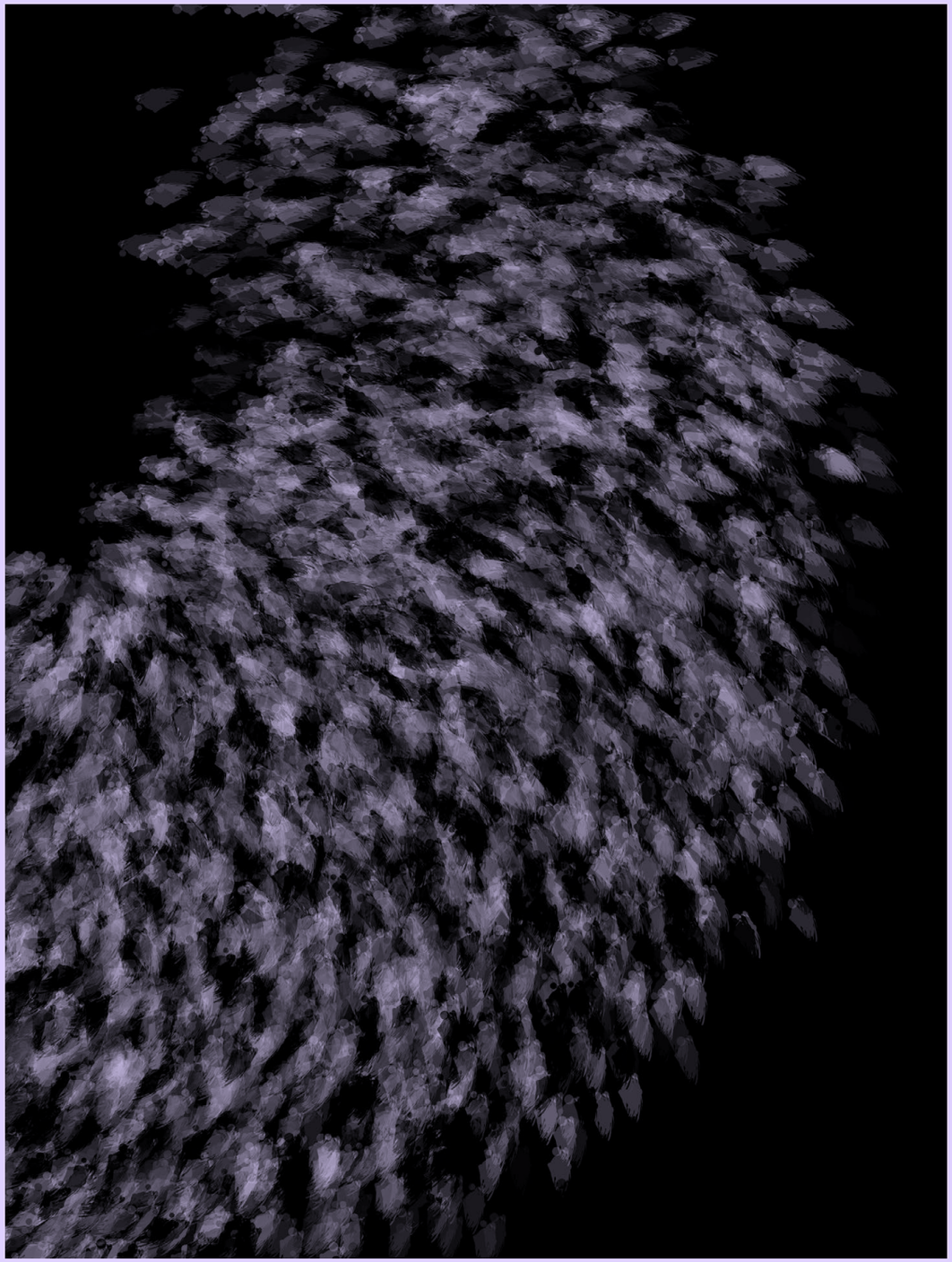
Above are some examples of the open source p5.js exercises I've developed to help teach creative coding over the years. Thank you for building such a wonderful software, organization, and a fun & welcoming environment!

— Matt DesLauriers (mattdesl)



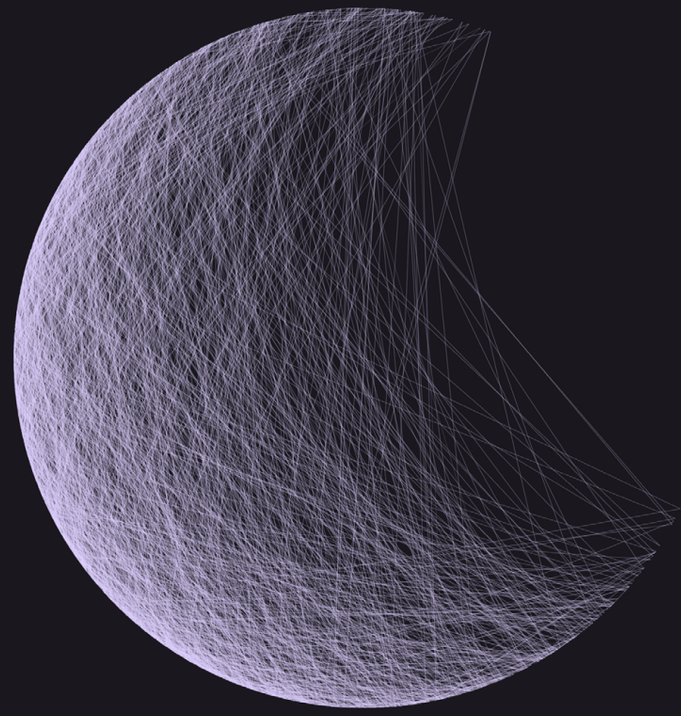








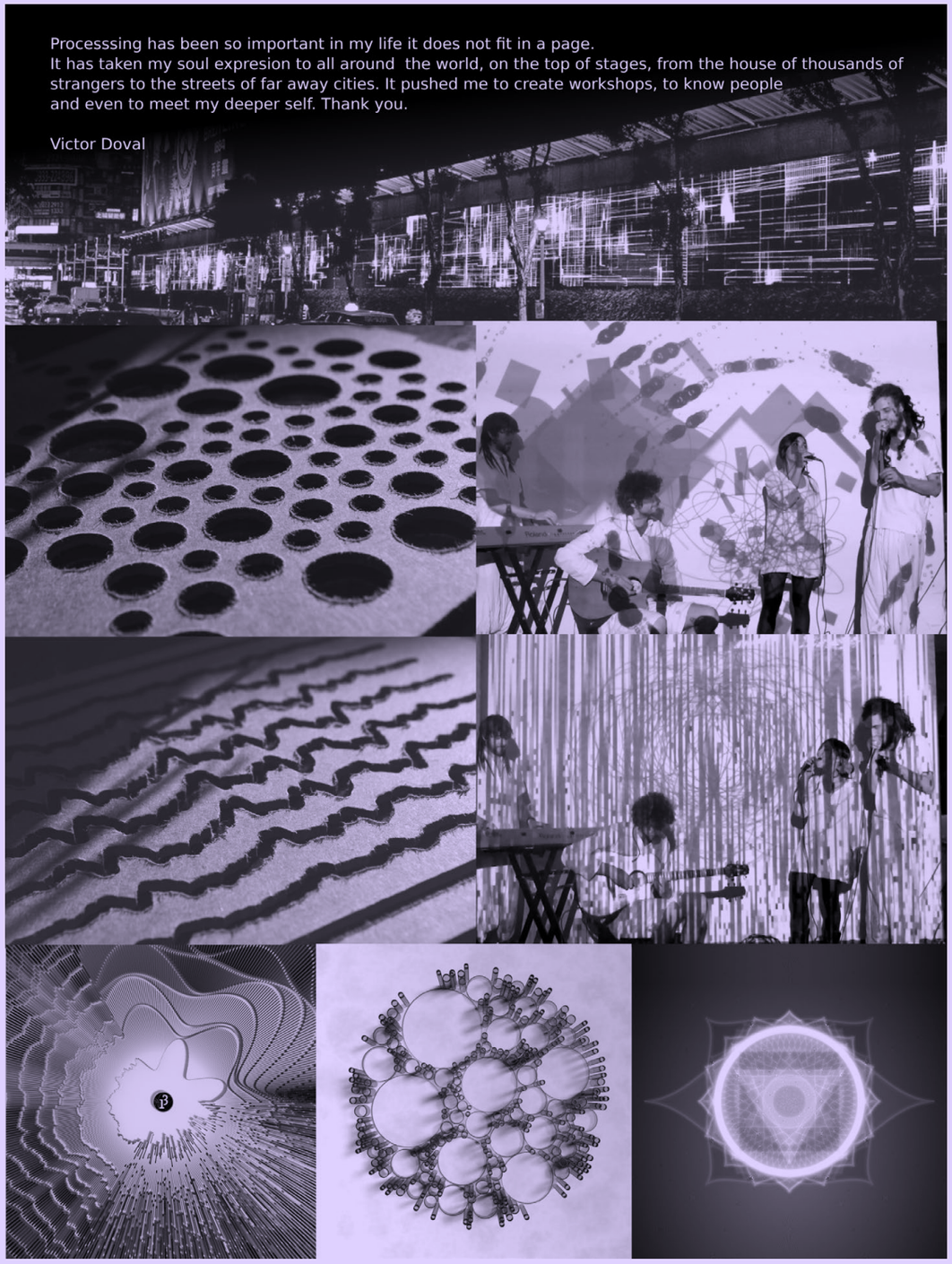
Happy 20th birthday Processing!
Discovering the magic of generative art was what sparked my passion for moving from architecture to software engineering.
Forever thank you!



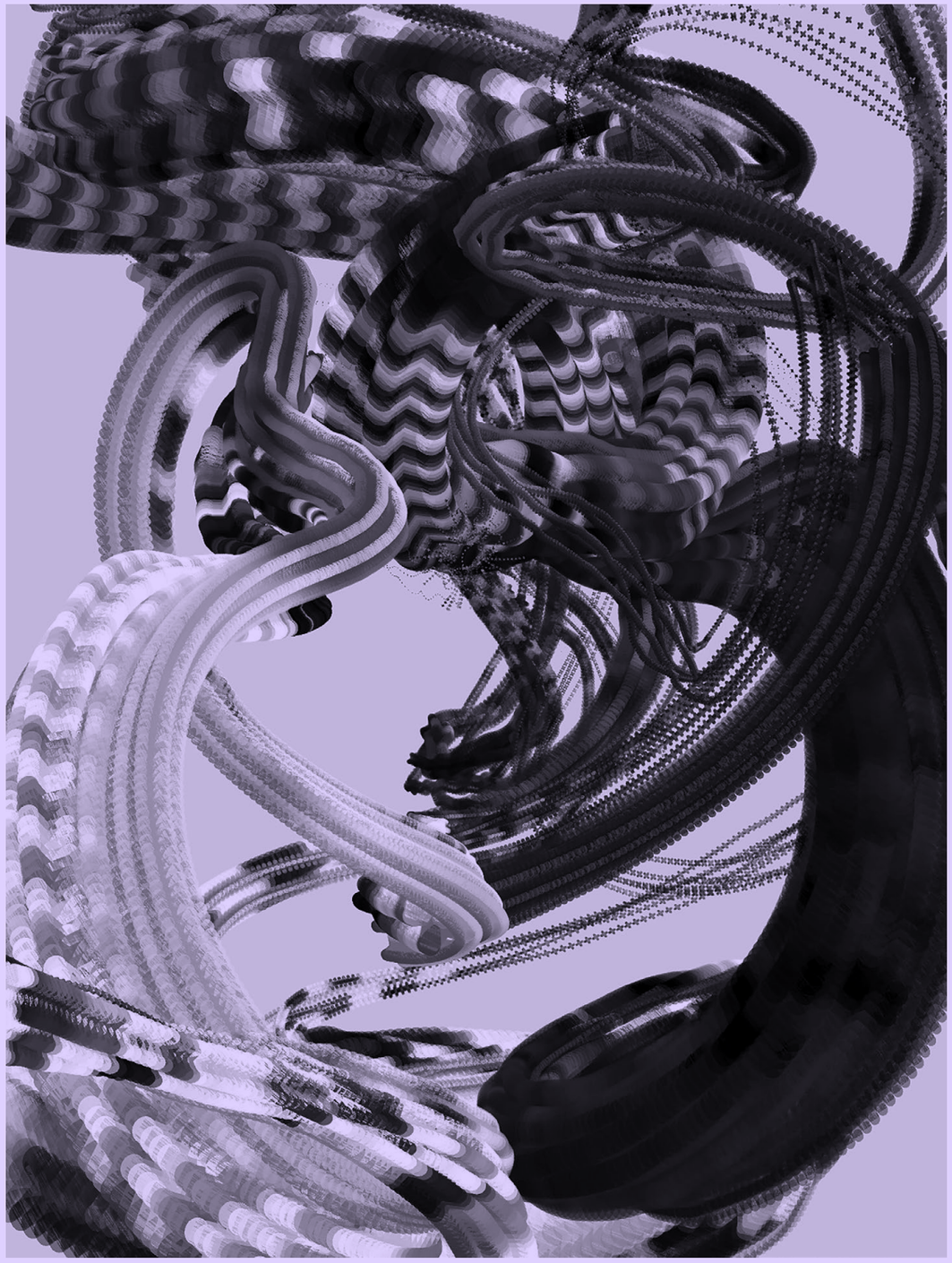
Andrea Diotallevi
Instagram: @andreadiotalleviart
Twitter: @a_diotallevi_

Processsing has been so important in my life it does not fit in a page.
It has taken my soul expresion to all around the world, on the top of stages, from the house of thousands of strangers to the streets of far away cities. It pushed me to create workshops, to know people and even to meet my deeper self. Thank you.

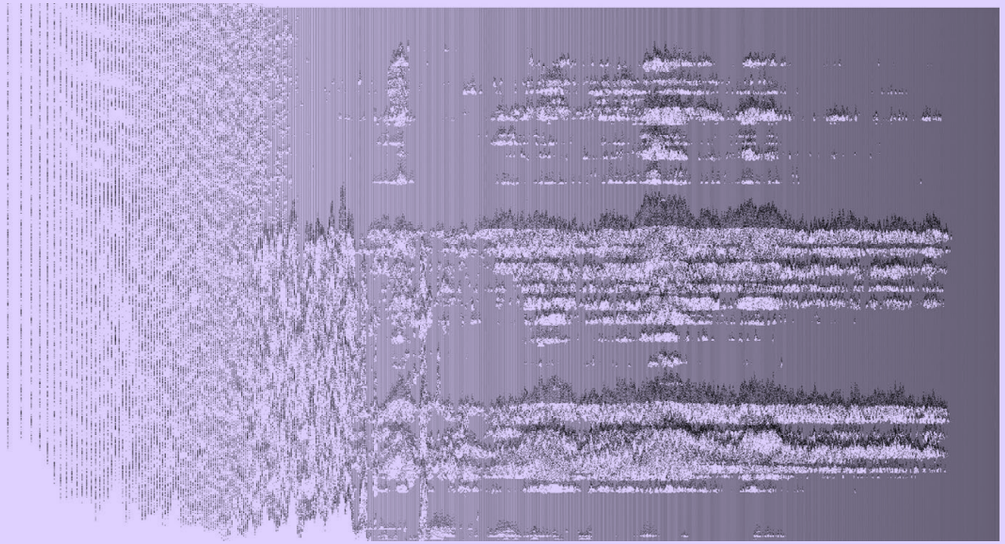
Victor Doval



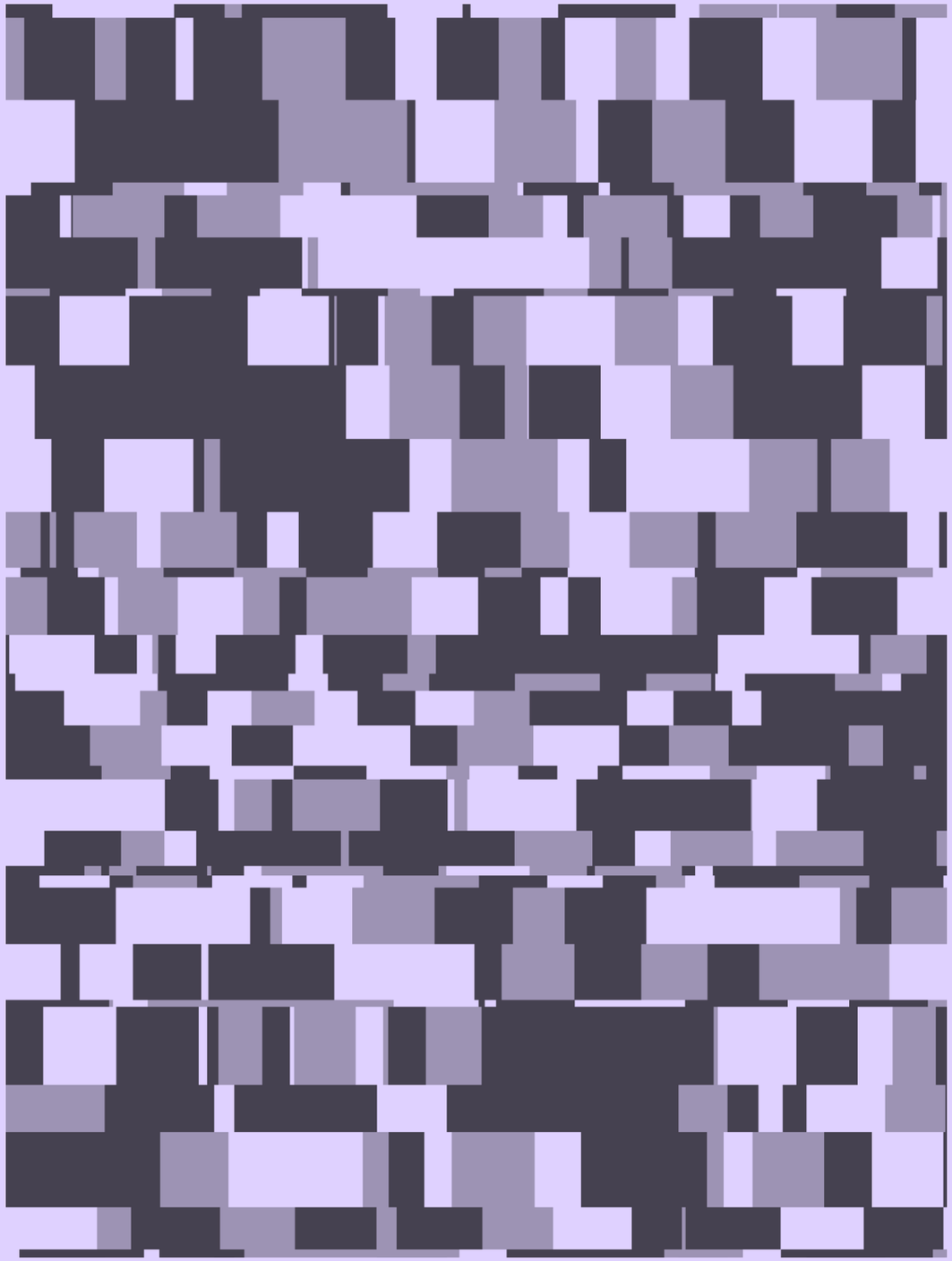


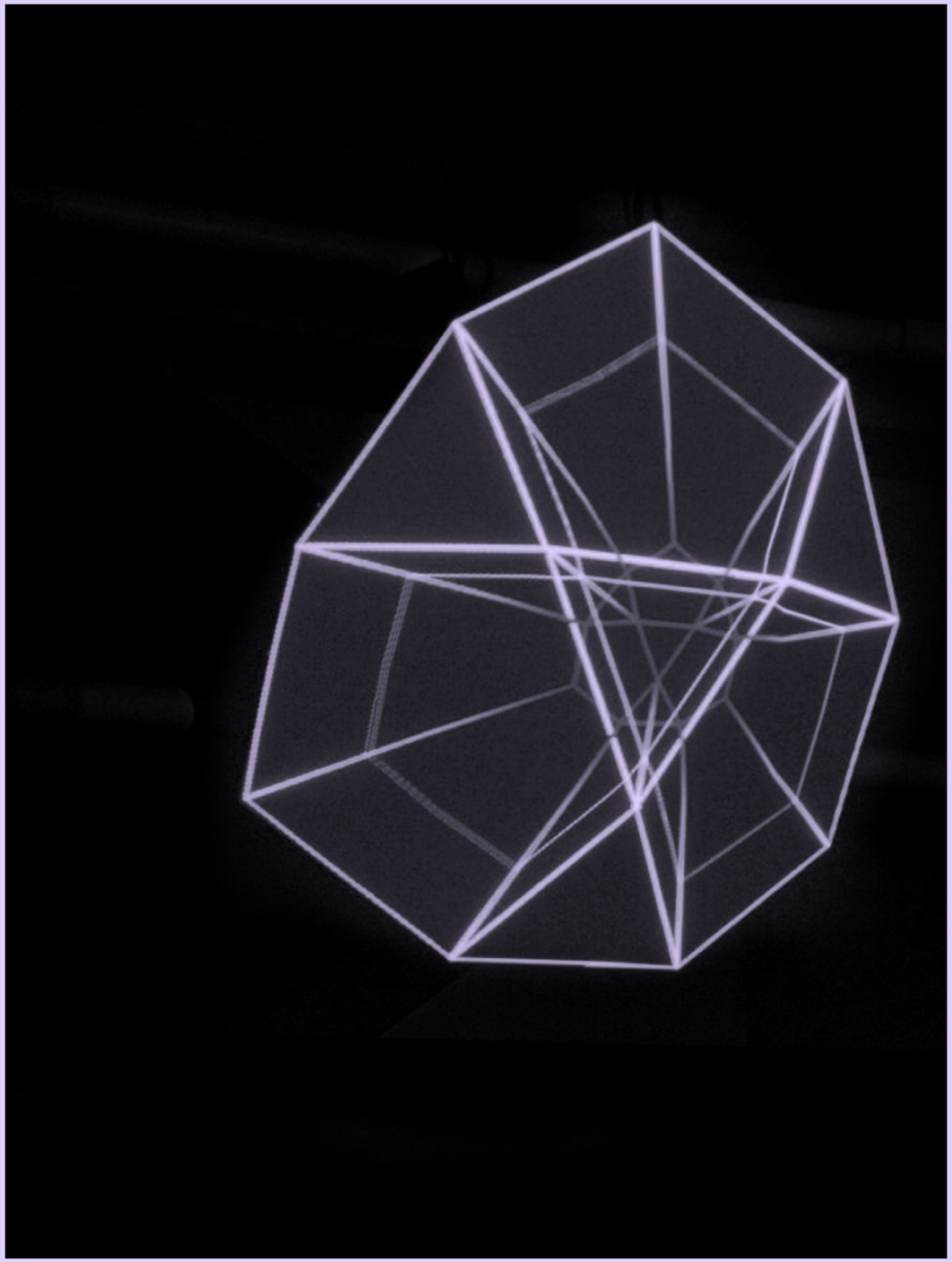


```
// love will tear us apart.
// rld, 2021.
//
// the fast fourier transform (FFT)
// jean-baptiste joseph fourier
// fourier theorem:
// a periodic signal may be expressed
// as the sum of a series of sine waves,
// each of which has a specific
// amplitude and phase.
//
// happy birthday, processing.
// https://openprocessing.org/sketch/1291541

var themic; // this is the audio input
var thefft; // this is an FFT object
var x = 0;
var ystep = 0;

function setup() {
        createCanvas(windowWidth, windowHeight);
        background(255);

        themic = new p5.AudioIn(); // connect to the microphone
        themic.start(); // turn on the microphone
        thefft = new p5.FFT(); // create an FFT object
        thefft.setInput(themic); // listen to the microphone

}

function draw() {
        stroke(0, 100);
        let thespectrum = thefft.analyze(); // load analysis data
        for(let i = 0;i<thespectrum.length;i++)
        {
                let x = sqrt(i/thespectrum.length) * width; // exponential frequency spacing
                let y = height - thespectrum[i]*1; // offset based on FFT channel amplitude
                point(x, y - ystep); // draw
        }
        ystep++; // shift waterfall scan line
        if(ystep>height) // top of screen
        {
                background(255); // clear
                ystep = 0; // reset scan line
        }
}
```

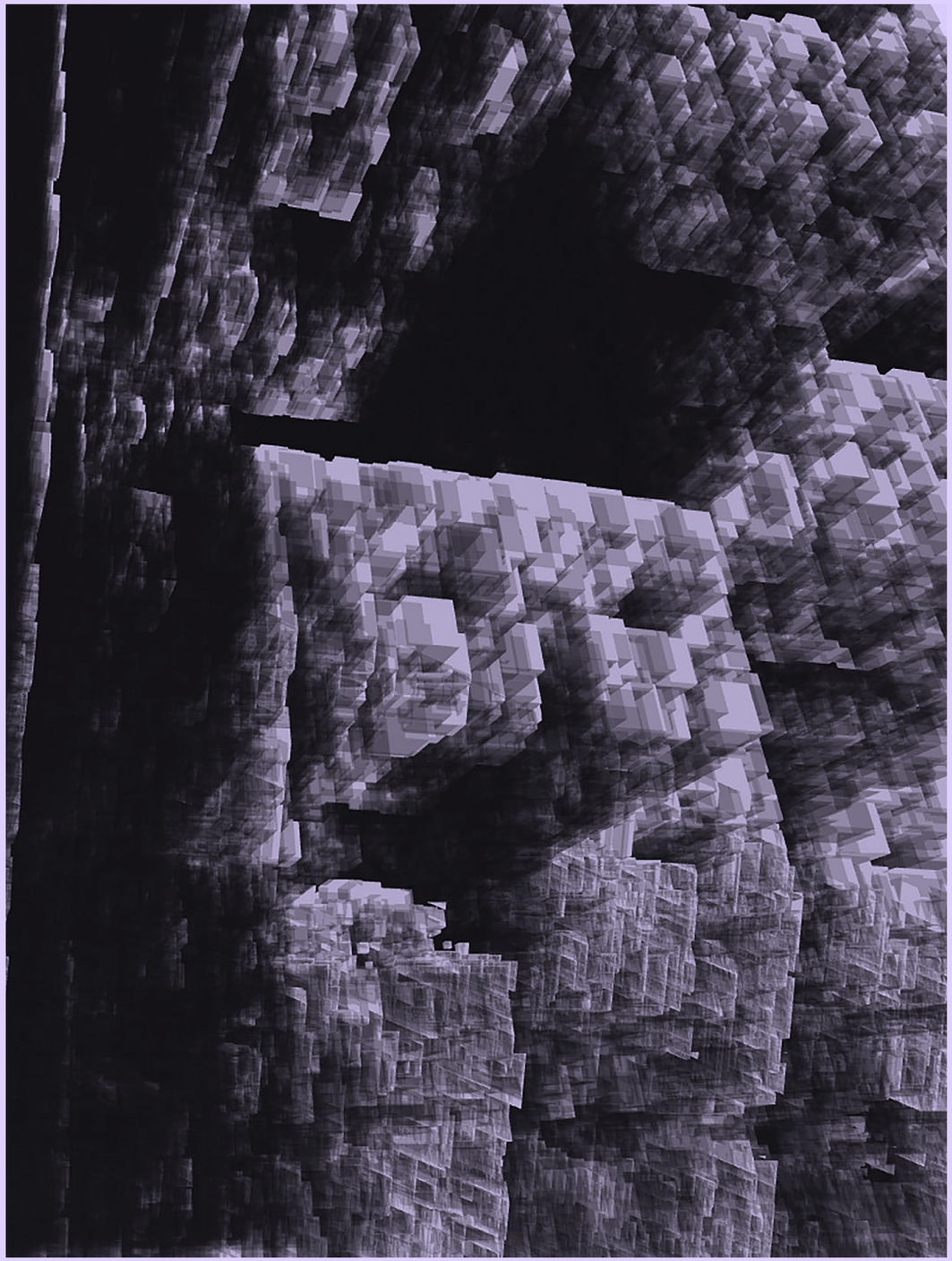


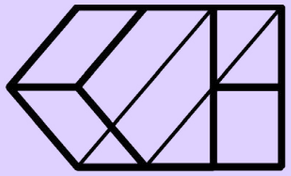




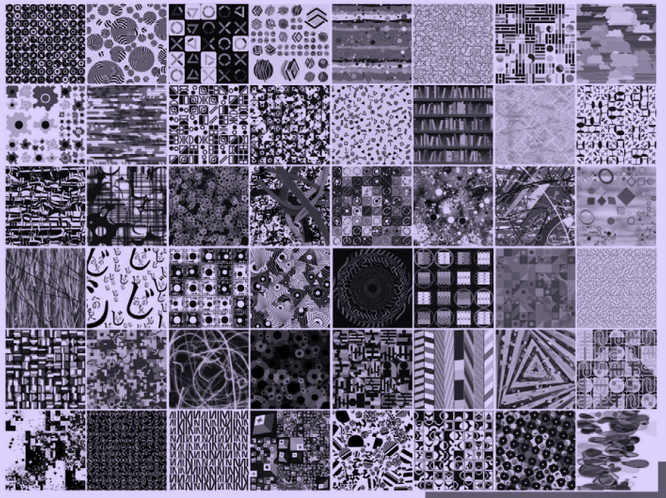








E.C.H / いしいえいいち
Artist, Creative Coder
https://eiichiishii-web.jimdofree.com/



"Daily Coding"
https://www.instagram.com/eikun_0903/

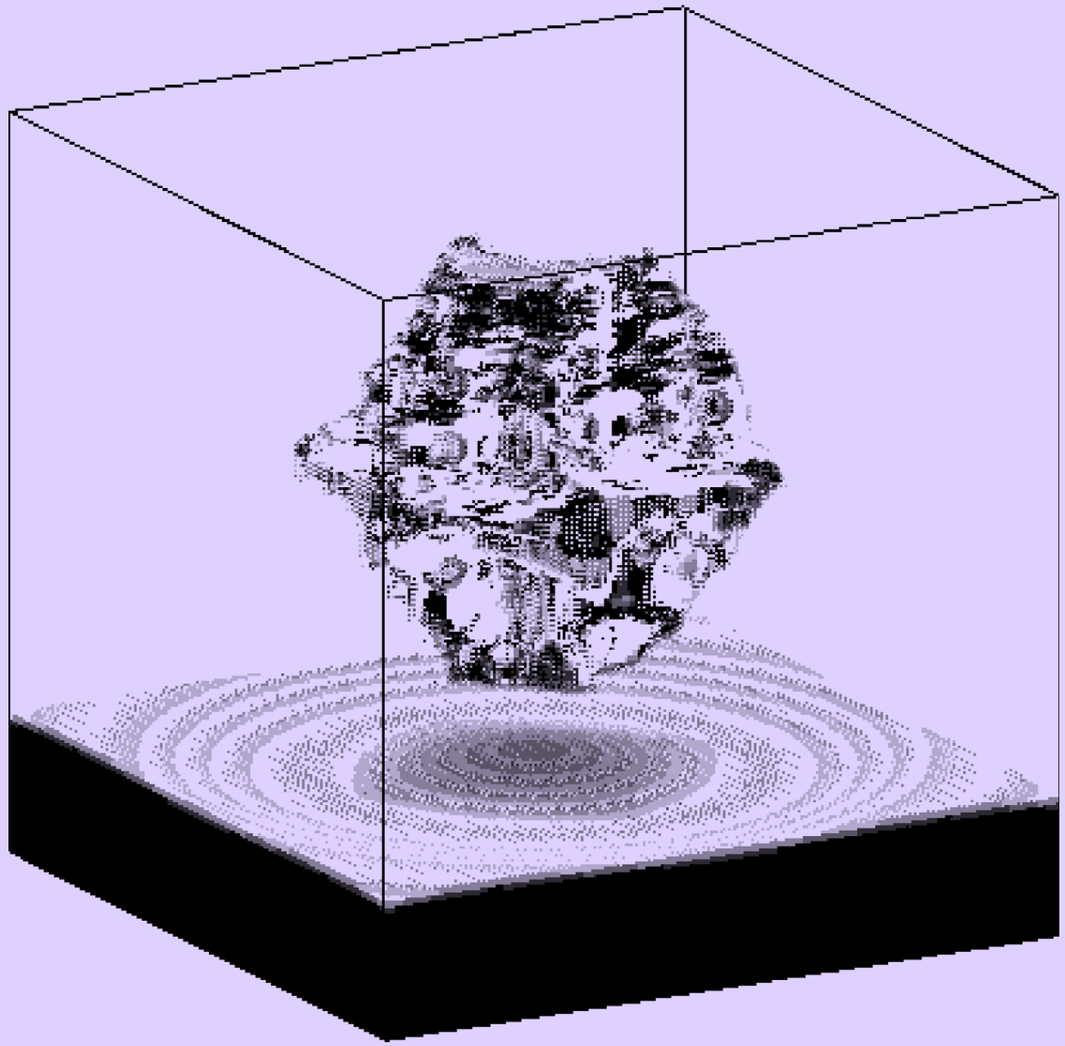
"Pastel"(Video Installation)

I started using Processing when I was in high school.
At the time, I was a member of the computer club, and when I was bored with general programming languages, I learned that there was a language suitable for creating graphics and digital art. That language was Processing.

I was immediately hooked on Processing because of its easy-to-understand grammar and ease of expression, even for beginners. Today, I am an artist, and I still use Processing and p5.js as my main tools for creation.

In recent years, the community has become more active, and this year, I learned that it has been 20 years since Processing was developed, and I am looking forward to its future development.








NullPointerException

Ursula Endlicher

## Power Line Feeds

**Interactive dinner table for two guests**

*Power Line Feeds, at Richmond Gallery, 2019. Produced at Laboratory, Interactive Residency, Spokane WA*

Power Line Feeds is a networked interactive multiple-media installation and dinner event inviting two guests at a time to enjoy a three-course meal served on a set of interactive dinnerware that visually communicates the dinner eating process of both diners via a screen mounted between them. Participants join into a "sensorial system theater", where eating and tasting, moving and drawing, interacting and communicating, become a simultaneously digital, visceral, and remote dining experience.

Inspired by the world-wide system of power lines the installation superimposes a variety of networked systems - electrical power, Internet, communication, and food - and turns the design of ceramic insulators, found on utility poles, into a three-part interactive custom-built dinnerware.

Each entrant's table manners are translated into "digital stains" - visual traces drawn by the move of their dishes, connected via motion sensors to the custom-built drawing software, and then projected onto the screen. Collected over the course of the event, a procedural map of all guests' eating behavior is created.

*Medium: Custom drawing software (Processing), interactive custom-built ceramic three-part dinnerware with built-in gyro- & accelerometer controllers, food, MacBook Pro, projector, rear projection screen, power line poles.*

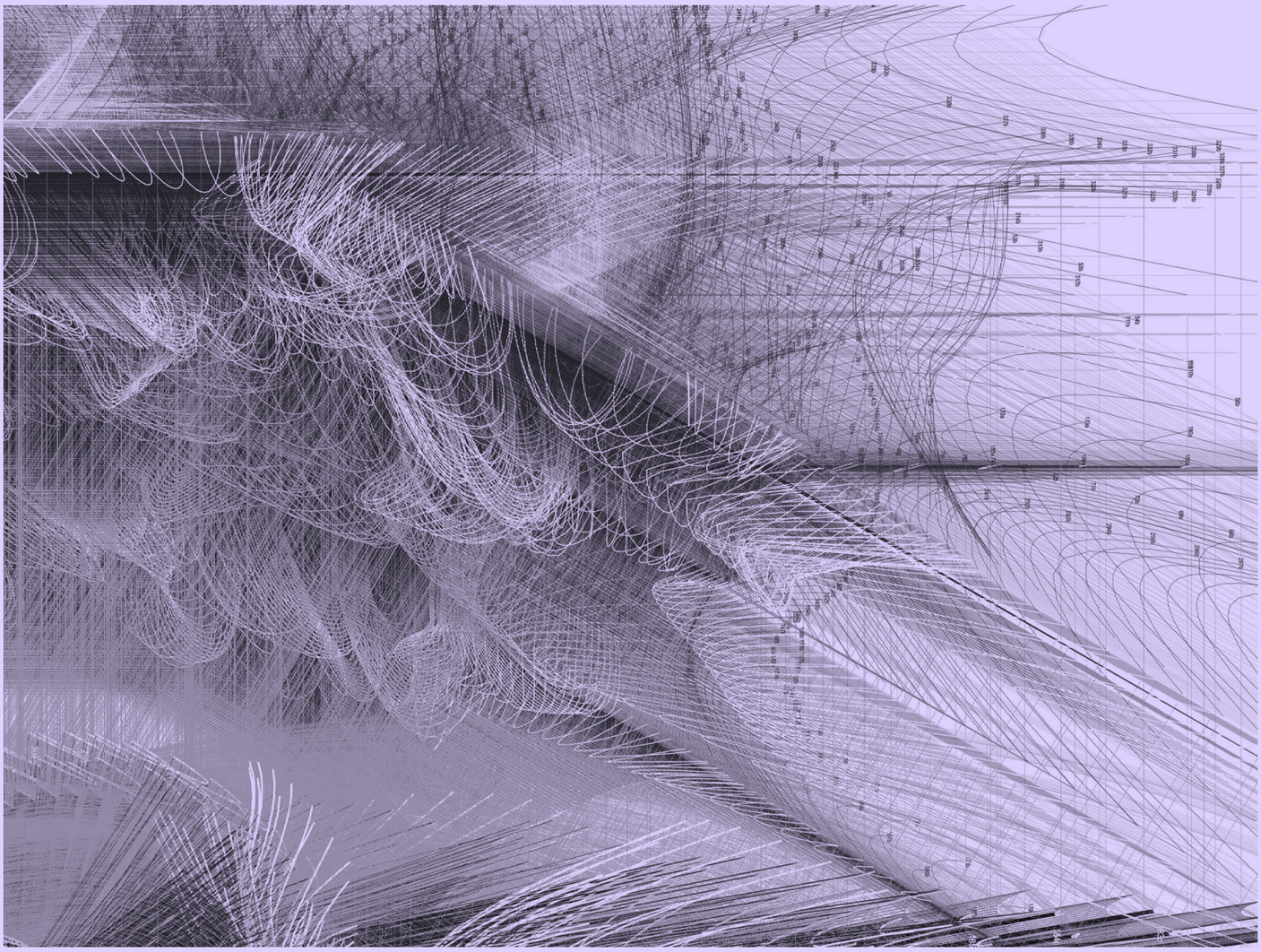
www.ursenal.net/PowerLineFeeds/



MARK ERICSON, @M_CERICSON

CON 155

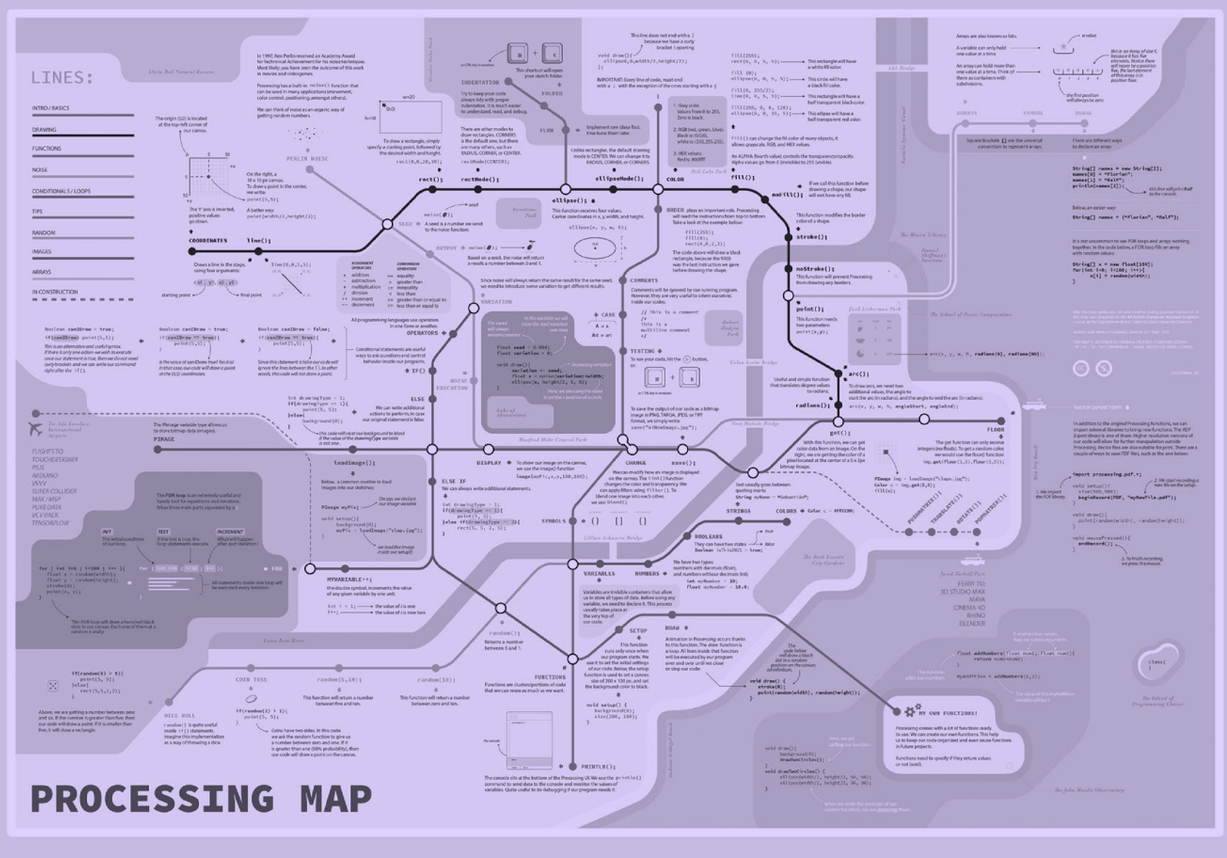
470

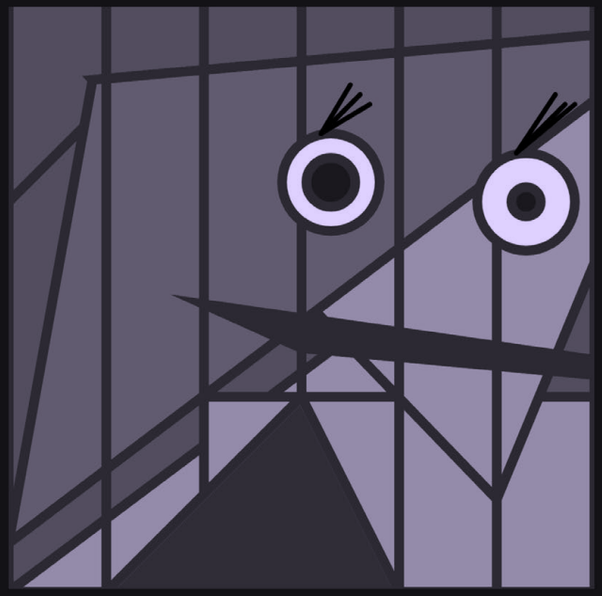


JM ESCALANTE

CON 156

471





# A Self-Portrait Through the Eyes of My Fiance

*By Ethan Omo*

For my first project in P5.js as part of the Emergent Digital Tools class at the University of Denver, I wanted to create something meaningful. "A Self-Portrait Through the Eyes of My Fiance" (2021) was born from a conversation with one of my close about her ADHD. She spoke about how difficult it was to find a partner that really understand how this manifestation of neurodivergence works.

This made me reflect on my interactions with my partner as she has ADHD.

As a result of my reflectionsl wanted to create a piece that reminds of what she experiences. My partner has told me she fixates on certain details like wild eyebrow hairs or eyes when she talks to people and loses track of their saying. This piece is meant to remind me that I need to consider her experience when I tell her things, or when she is having trouble with a task. It is a reminder for to always try to understand better and grow into a better partner.

The coding behind the piece is relatively simple. It based around the idea of just using 2d primitives to construct a portrait. The colors and the abstract style are intentional.

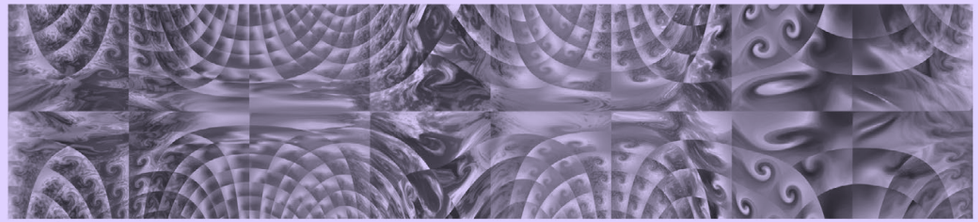
Overall, this was a great introduction to p5.js. and the community has been a great resource when it comes me trying to learn my first primary coding language. Thank you p5.js community!

Link: to the editble file in p5.js: ***https://editor.p5js.org/eomo9106/sketches/5FmYPQBk8***

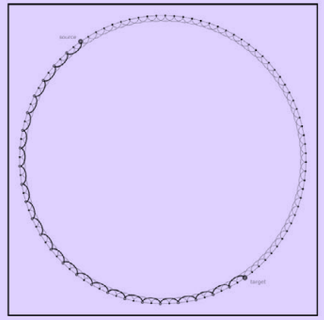
And for fun here is the code for the eyes:

```
fill(255, 255, 255);
  circle(330, 180, 100);
  circle(530, 200, 100);
  fill(74, 71, 18);
  circle(530, 200, 30);
  circle(330, 180, 50);

  stroke(0);
  strokeWeight(5);
  line(350, 80, 320, 130,);
  line(360, 90, 320, 130,);
  line(370, 100, 320, 130,);
  line(560, 90, 520, 150,);
  line(570, 100, 520, 150,);
  line(580, 100, 520, 150,);
```

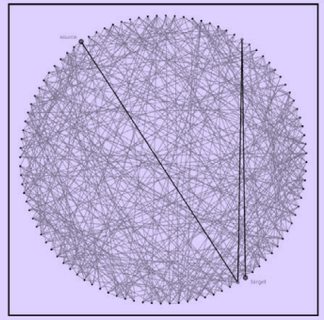




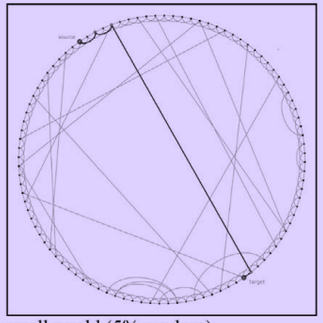
*salia* (music score)



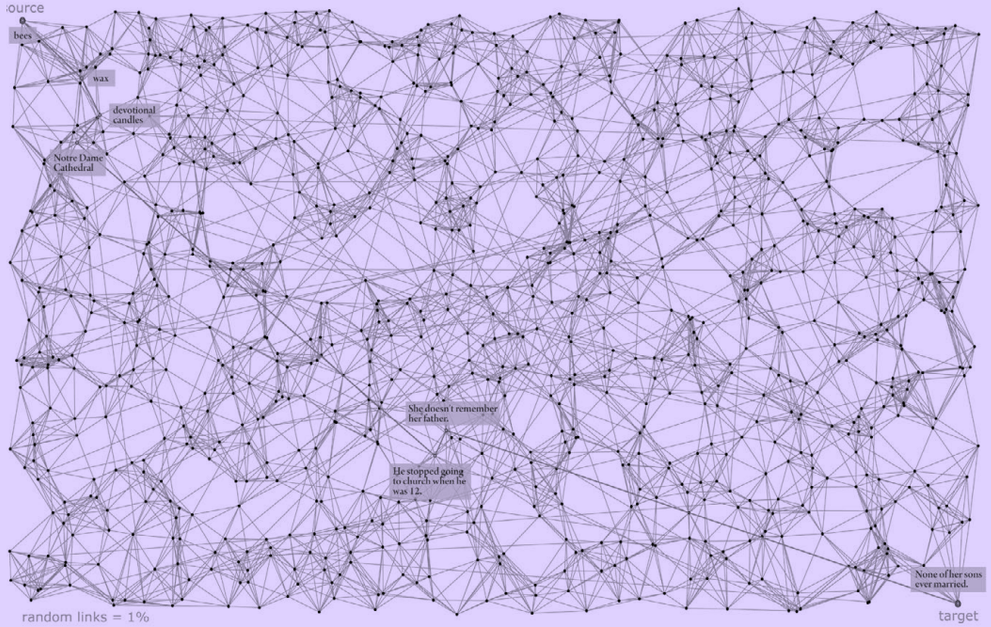
clustered network



random network



small world (5% random)




*melancholia*

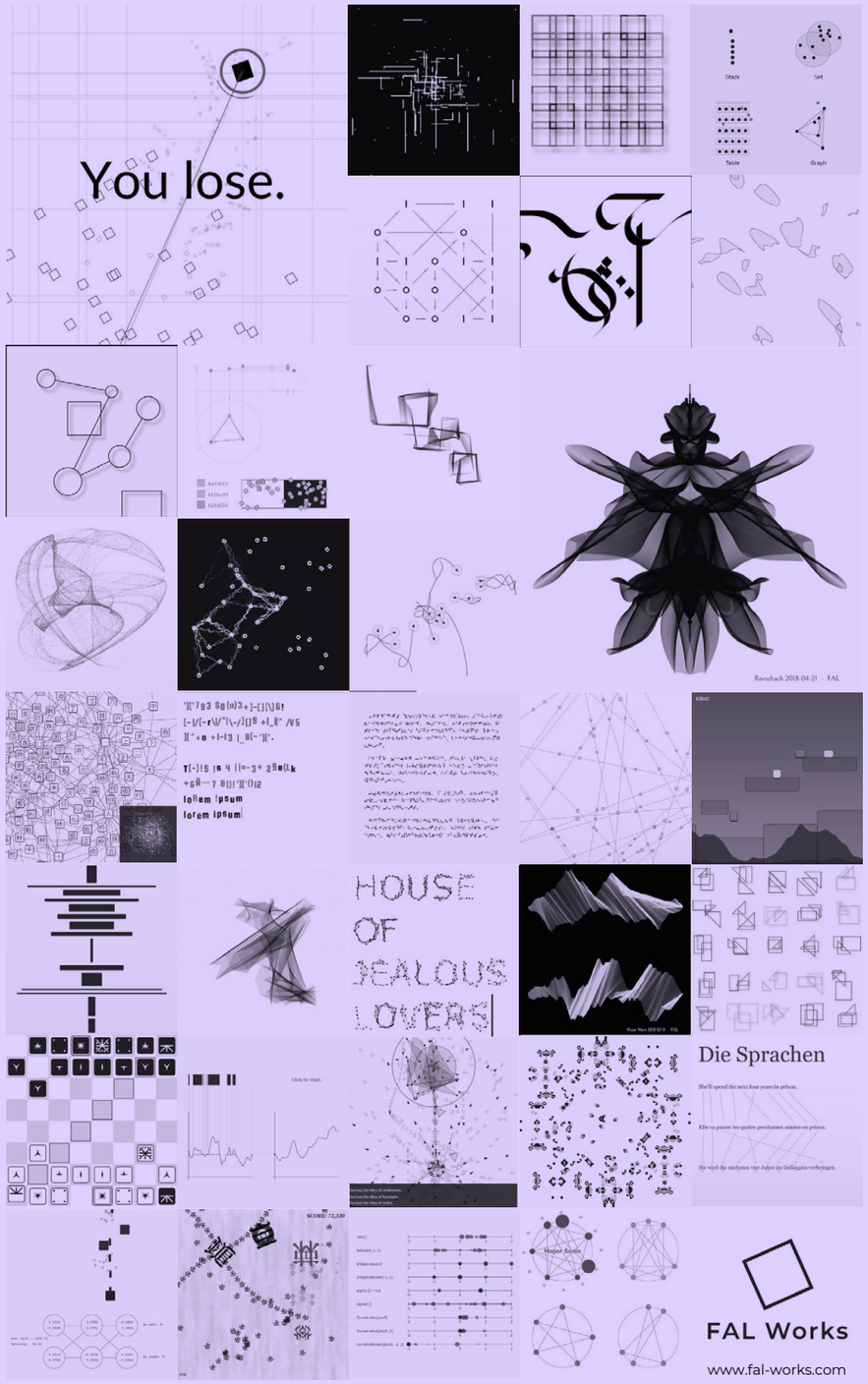
Parsing a network from source to target, is all about finding our way—from problem to solution, from person to person, from here to home. We find our way through a small world, a network of incestuous links and a little randomness. It's pathfinding through myriad maps. And as behavior follows structure, it's not surprising that the intricacies of our lives mirror the filagreed arbors of our neural forests.

I am very grateful to be part of a community that is inclusive, creative, open, accessible and caring. The stability and longevity of that outlives any one digital file. Thank you to the Processing leaders and community members, today and always.

evhan55

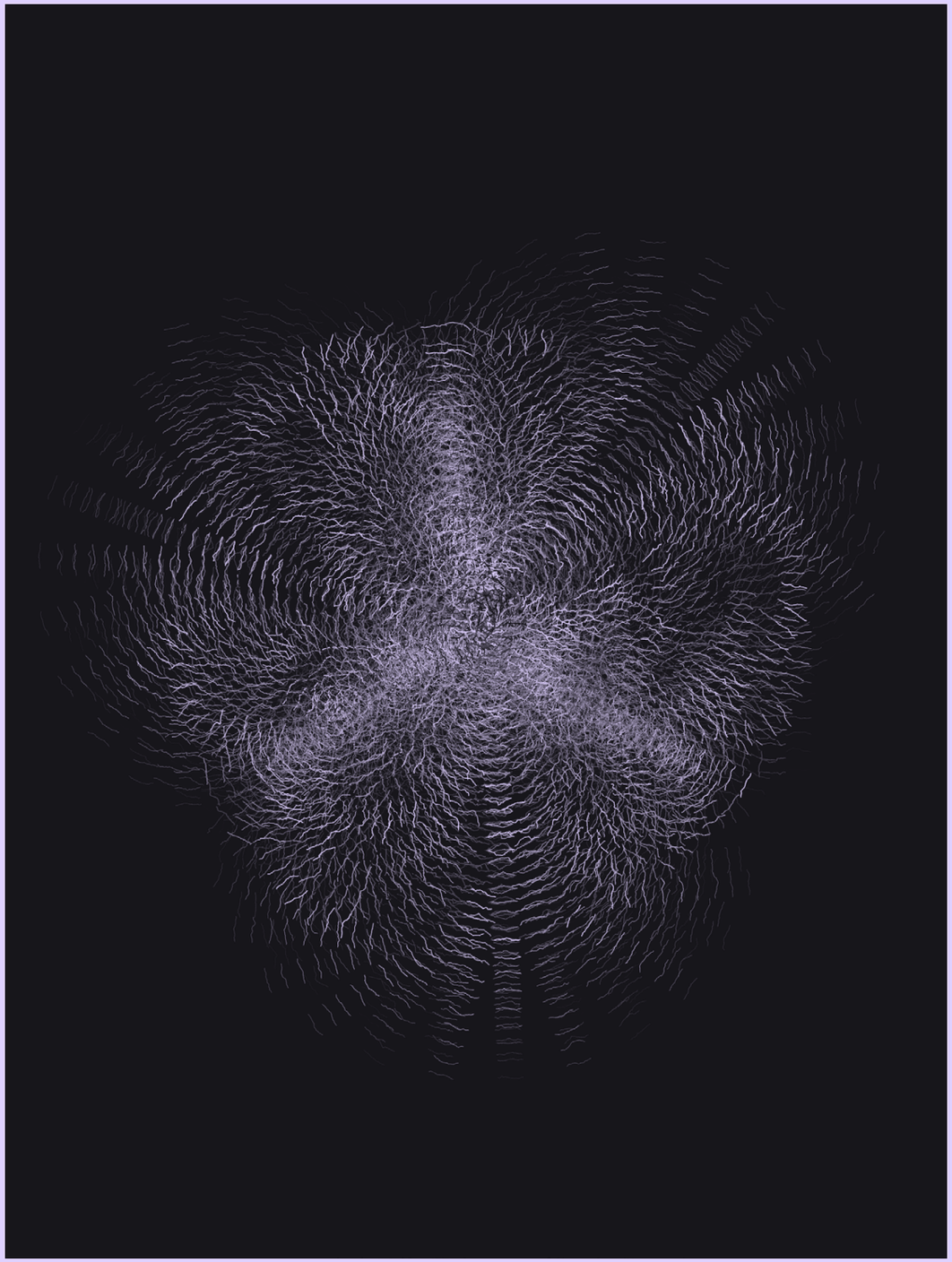


of histories
present and near
we are the world,
[/]
ubiquitous artific
meta materials re
Cover Me
of the human pre
Kindness
eroticism & tears

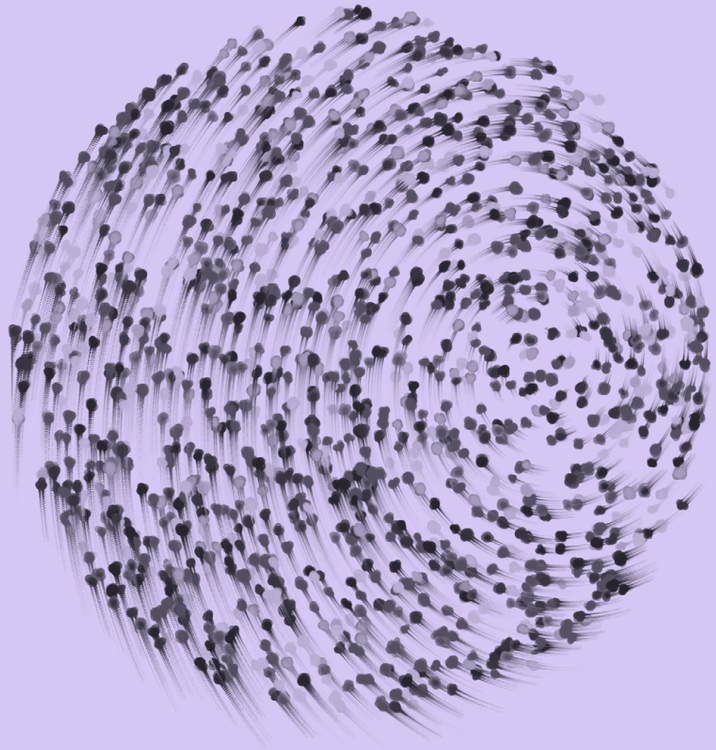

You lose.
Stack
Set
Table
Graph
HOUSE
OF
JEALOUS
LOVERS
Die Sprachen
FAL Works
www.fal-works.com




Can the subaltern speak?
-.-. .- -. / - .... . / ... ..- -... .- .-.. - . .-. -. / ... .--. . . .- -.- ..--..
آیا فرودست مي تواند سخن بگويد؟
C a n t h e S u b a l t e r n S p e a k ?
Can: [ ]
an : [t]
n t: [h]
th: [e]
the: [ ]
he : [S]
e S: [u]
Su: [b]
Sub: [a]
uba: [l]
bal: [t]
alt: [e]
lte: [r]
ter: [n]
ern: [ ]
rn : [S]
n S: [p]
Sp: [e]
Spe: [a]
pea: [k]



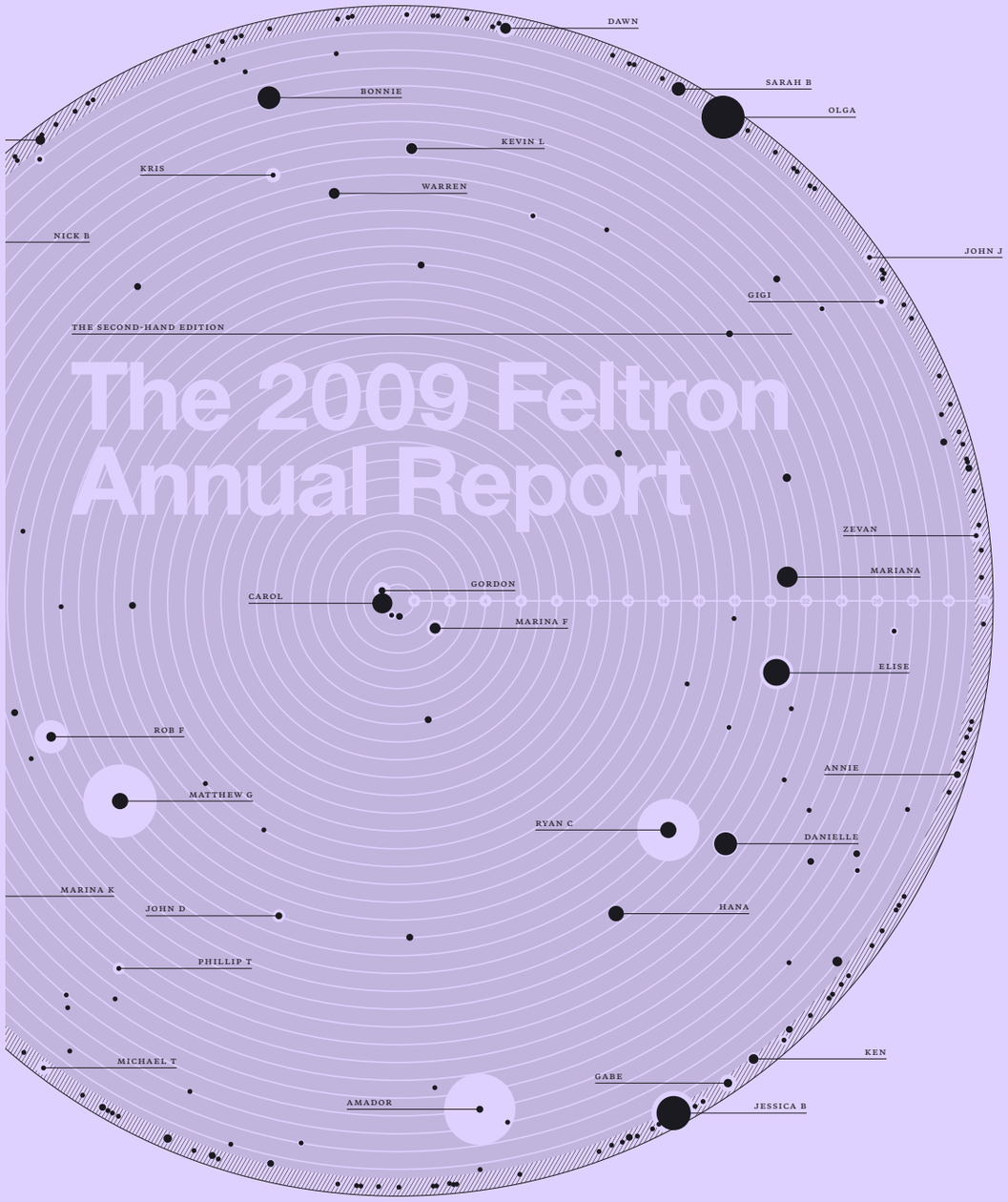




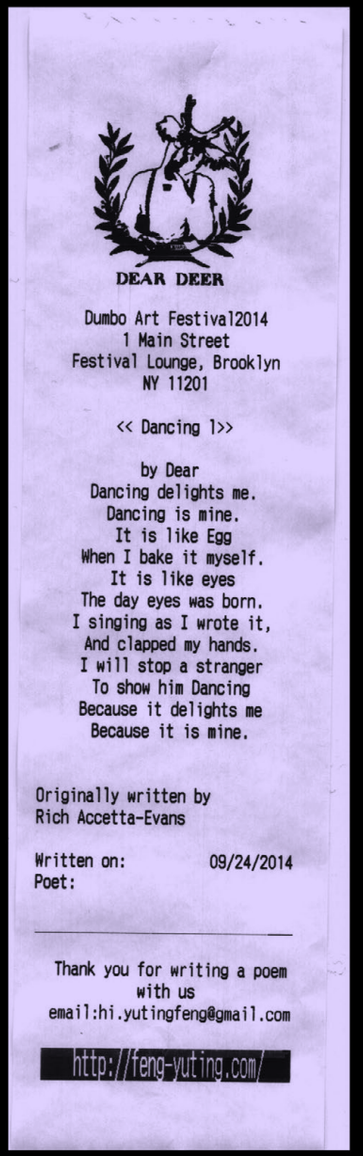
LOVEBOMB

**lovebomb (noun):** a handheld contraption built of thoughts, feelings, and idiosyncrasies, that initially smites one and eventually kills them with love.

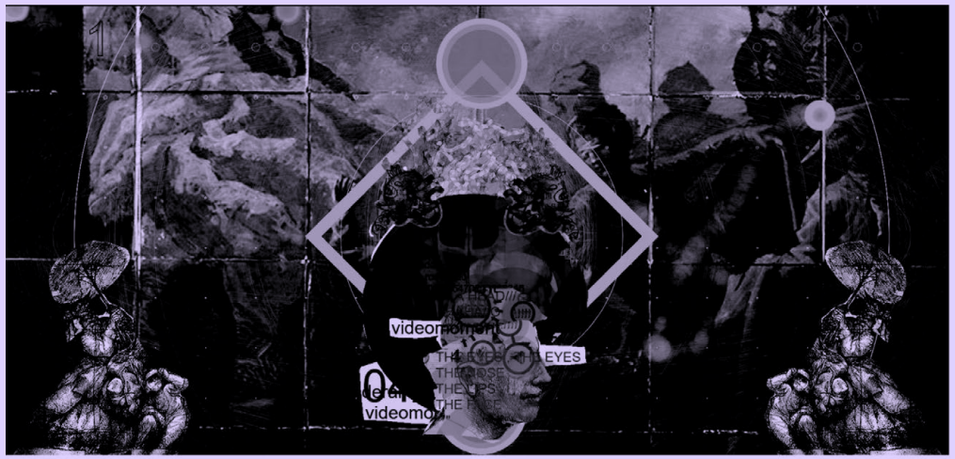
**usage:** She accidentally dropped a lovebomb in the midst of an unsuspecting crowd. Fortunately, most of the people escaped unscathed, however there was one casualty.

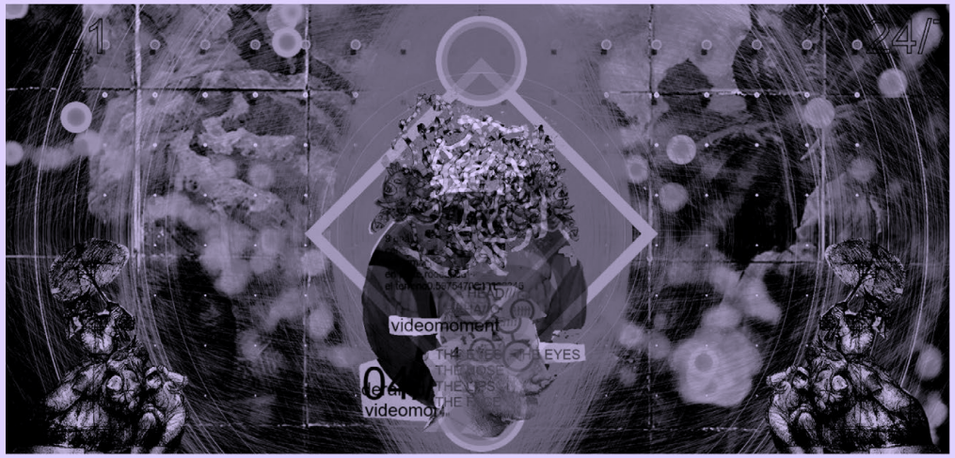


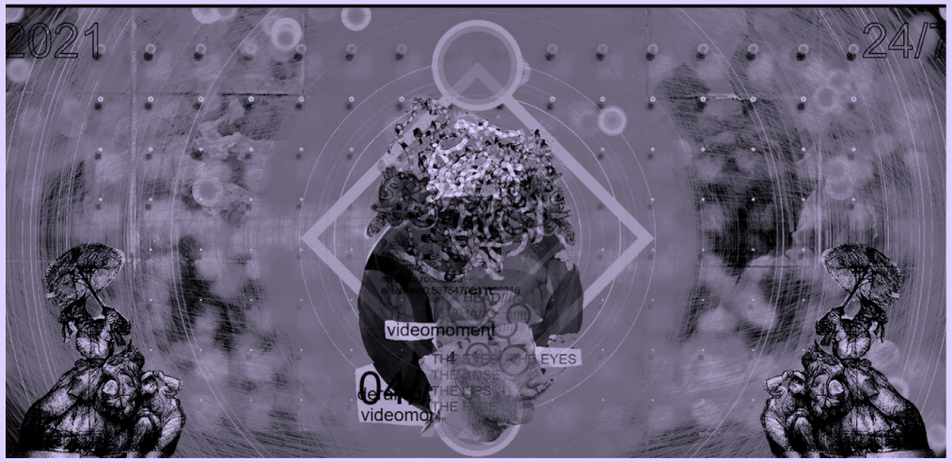


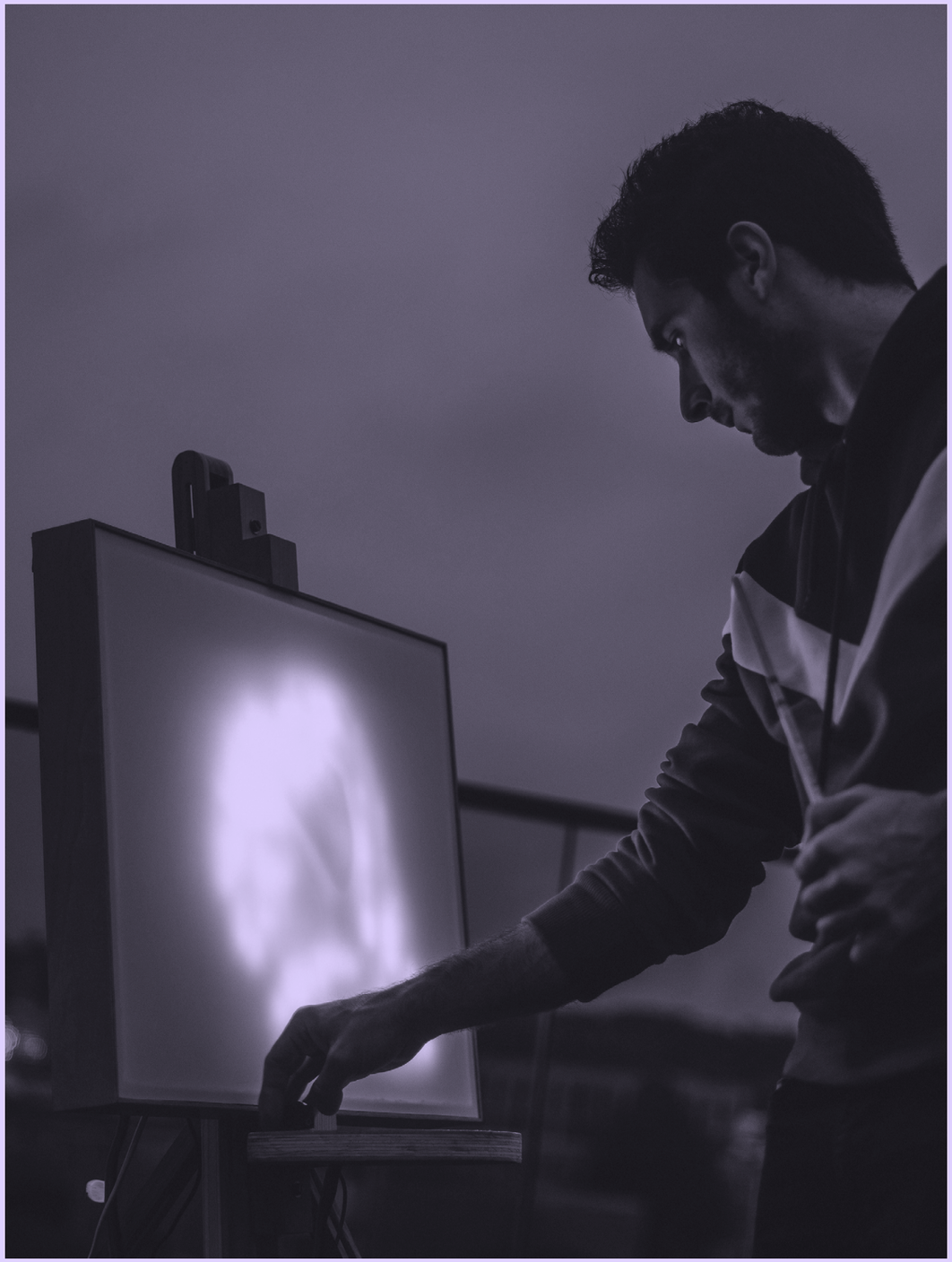

There are more than 5,000 poems created and printed with Dear Deer.


videomoment
EYES


videomoment
EYES


videomoment
EYES





LUIS FERREIRA (@SCHUUR.CREATIONS) CON 169 484

NICK FOX-GIEG CON 170 485

## *Lüme (2013)*

**Elizabeth Bigger and Luis Fraguada**

Lüme is an electronically infused clothing collection which integrates dynamic, user customizable elements driven wirelessly from a common mobile phone.

The Lüme collection won the Jury Prize in the Aesthetic Category at the 17th International Symposium of Wearable Computers (ISWC) in 2013.

Processing was used to develop an Android app which controlled the garment color via bluetooth.

https://datable.net
elizabeth@datable.net
luis@datable.net



# Sizecoding

## *Tiny Executable Art*

Sizecoding is the art of creating a very tiny program for a specific computing platform. These programs are made in the assembly language for the processor (CPU) used by the system. The 6502 was a popular CPU in the home computers from the 80s, e.g. Apple, Atari, Commodore and BBC. There is still a fanatic community for these old systems which nowadays we call retro computing.

## *Demoscene*

There are sizecoding competitions organised within the computer art subculture called “demoscene”. Demos are usually categorised by the allowed size of the executable program. The tiny size categories are 256 bytes, 128 bytes or sometimes even smaller.

## DSYNC // 32 Bytes

DSYNC or “Diagonal Synchronisation” is a play on the technical term HSYNC (Horizontal Synchronisation), a signal to tell the computer monitor to stop drawing the current horizontal line and start the next. Using exact timing a diagonal color pattern is produced. The program runs on an Atari XL/XE home computer.

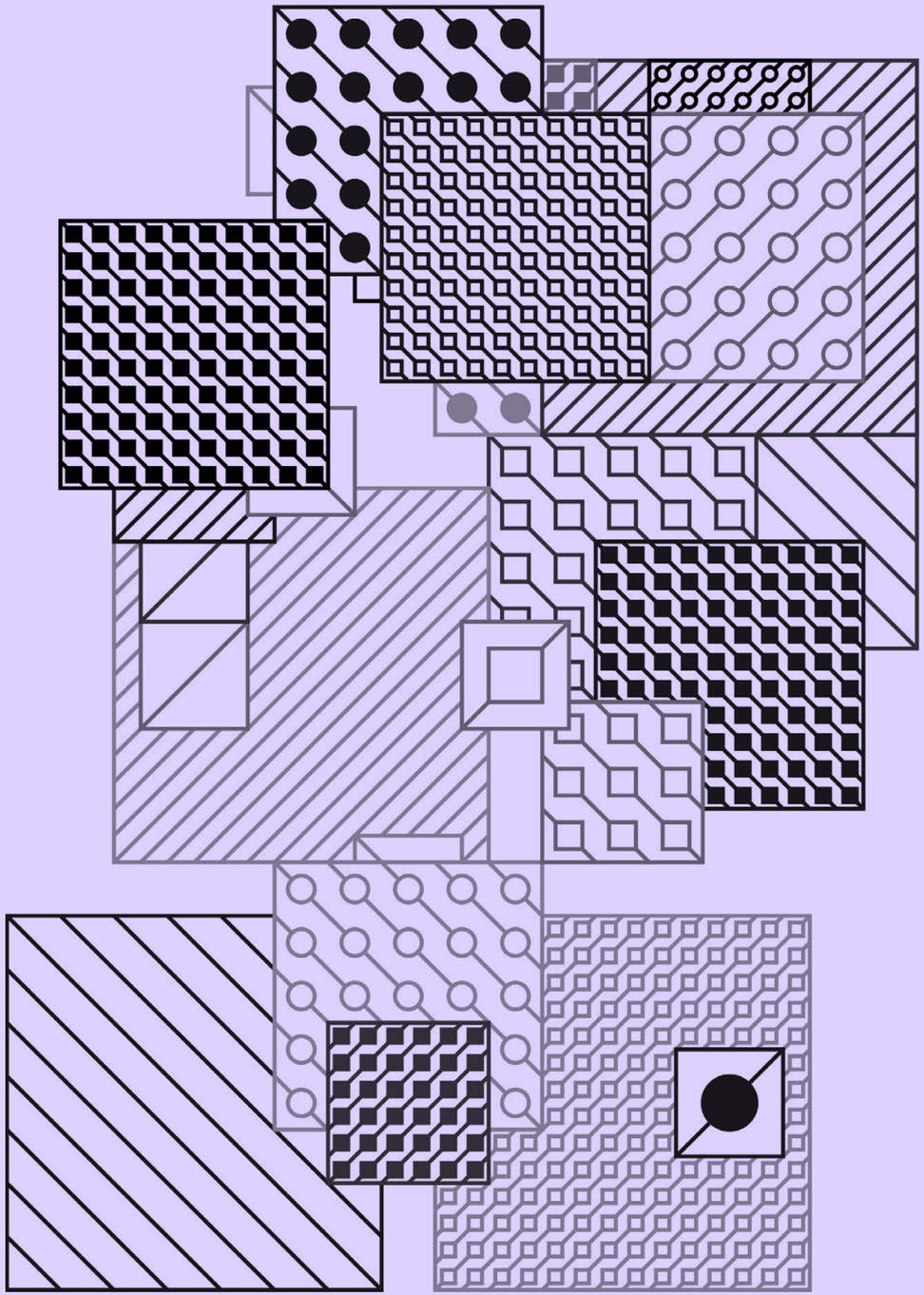
6502 Machine code

```
A2 00 8E 2F 02 A9 A8 8D
01 D2 8D 00 D2 29 F0 8D
1A D0 AD 0B D4 65 14 8D
1A D0 18 90 ED
```

6502 Assembly code

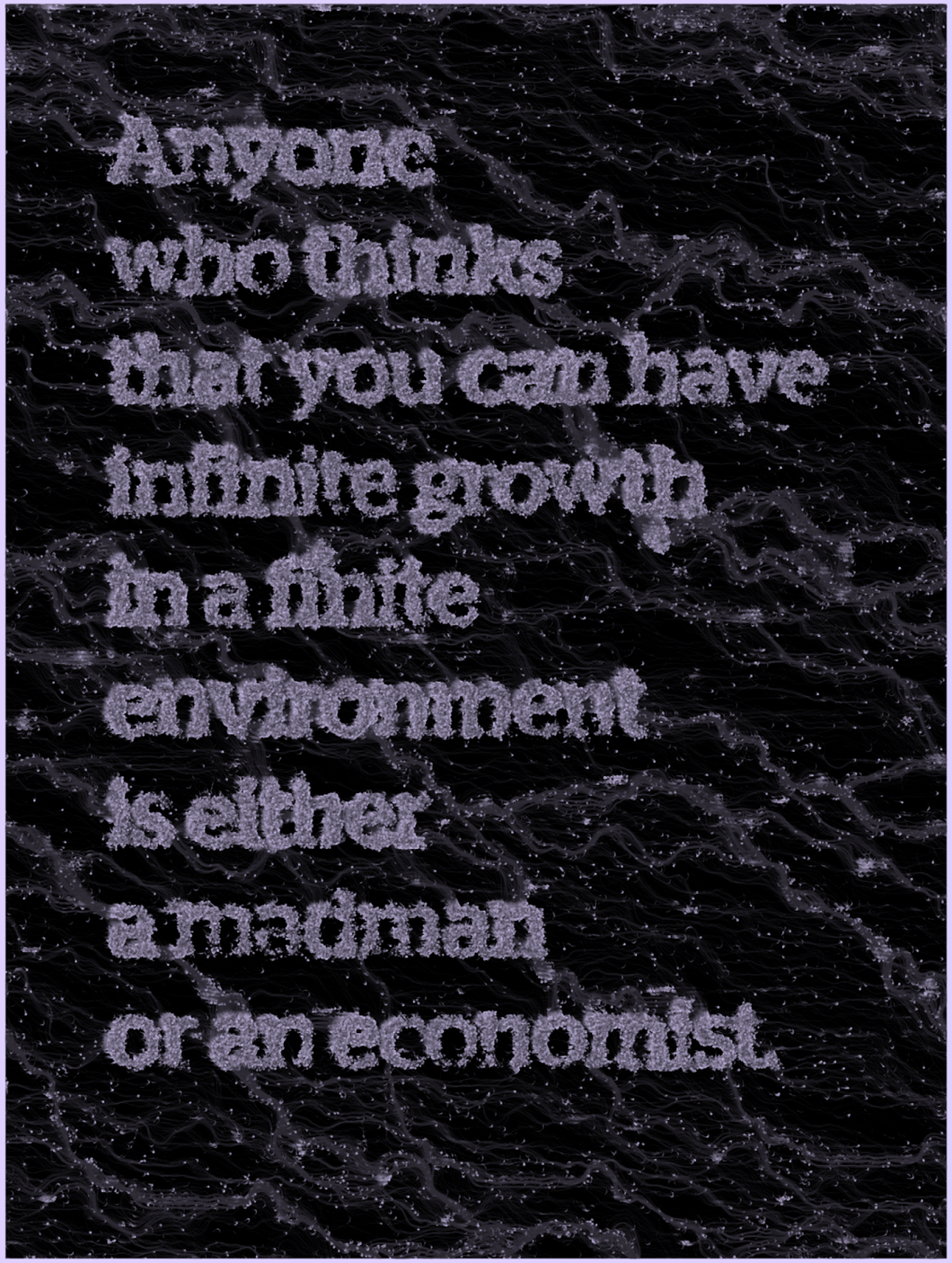
```
    ldx #0       ; x = 0
    stx 559      ; DMA off
    lda #$a8     ; audio control
    sta $d201    ; pure tones, volume 8
colors
    sta $d200    ; audio frequency = a
    and #$f0     ; filter color bits
    sta $d01a    ; store a in background color
    lda $d40b    ; a = vertical line counter
    adc 20       ; a = a + frame counter
    sta $d01a    ; store a in background color
    clc          ; c = clear
    bcc colors   ; jump to colors
```

F#READY, 2021-09-11
GitHub: https://github.com/FreddyOffenga
Demozoo: https://demozoo.org/sceners/35273/

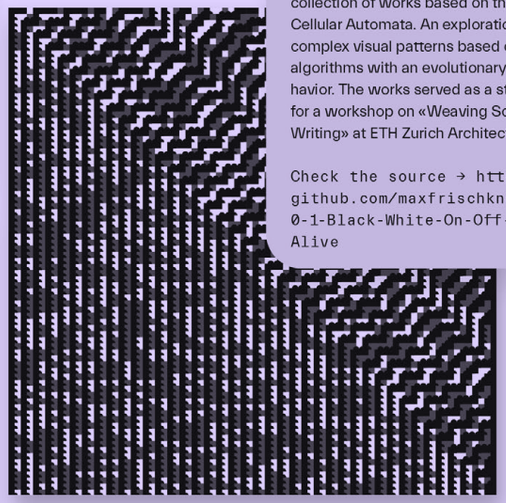


SASKIA FREEKE



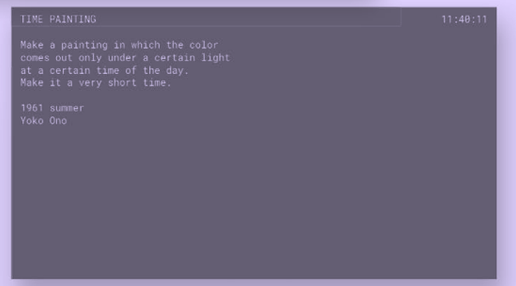


Processing was my first contact with code. And learning to code helped me to understand how digitization shapes our environment. How it influences our mental and material realities. And it allowed me to get politically and socially involved in today's digital society. Without the Processing Community, none of this would have been possible. 🙏🙏🙏



**0/1 Black/White On/Off Dead/Alive** is a collection of works based on the concept of Cellular Automata. An exploration of complex visual patterns based on simple algorithms with an evolutionary behavior. The works served as a starting point for a workshop on «Weaving Scripting Writing» at ETH Zurich Architecture 2019.

Check the source → https://github.com/maxfrischknecht/0-1-Black-White-On-Off-Dead-Alive




**Time Code Time Painting** follows Yoko Ono's instructions and creates a computational drawing that is constantly present but only visible at a certain point during the day. To illustrate the changing light conditions, the color of the painting is evolving over a 24h cycle showing 360 different colors.

Can you find the drawing? → https://maxfrischknecht.github.io/Time_Code_Time_Painting/Code_Time_Painting

**Allegoria** is a series of experimental video installations created in collaboration with fashion designer Anja Bodenmann. The film footage taken during a photoshoot was subsequently alienated Processing's noise function.



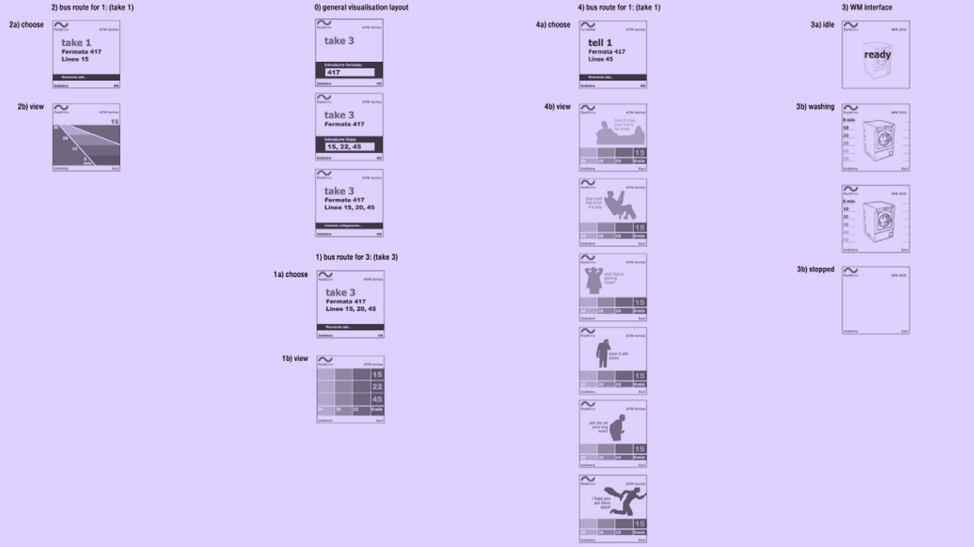
Collab? Say hi!
https://maxfrischknecht.ch/



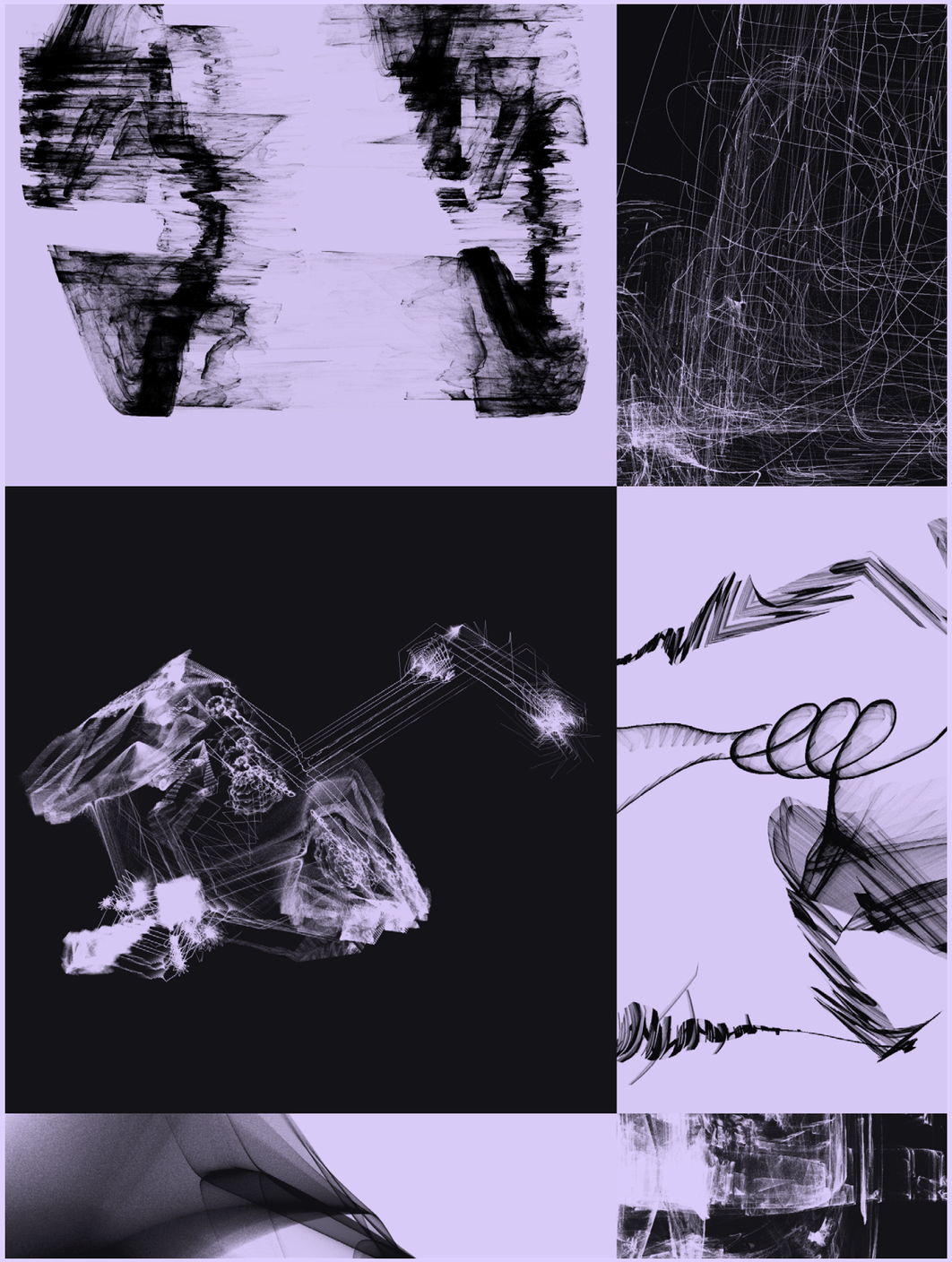
# fluidtime apps

take 1 take 3 tell 1 wm

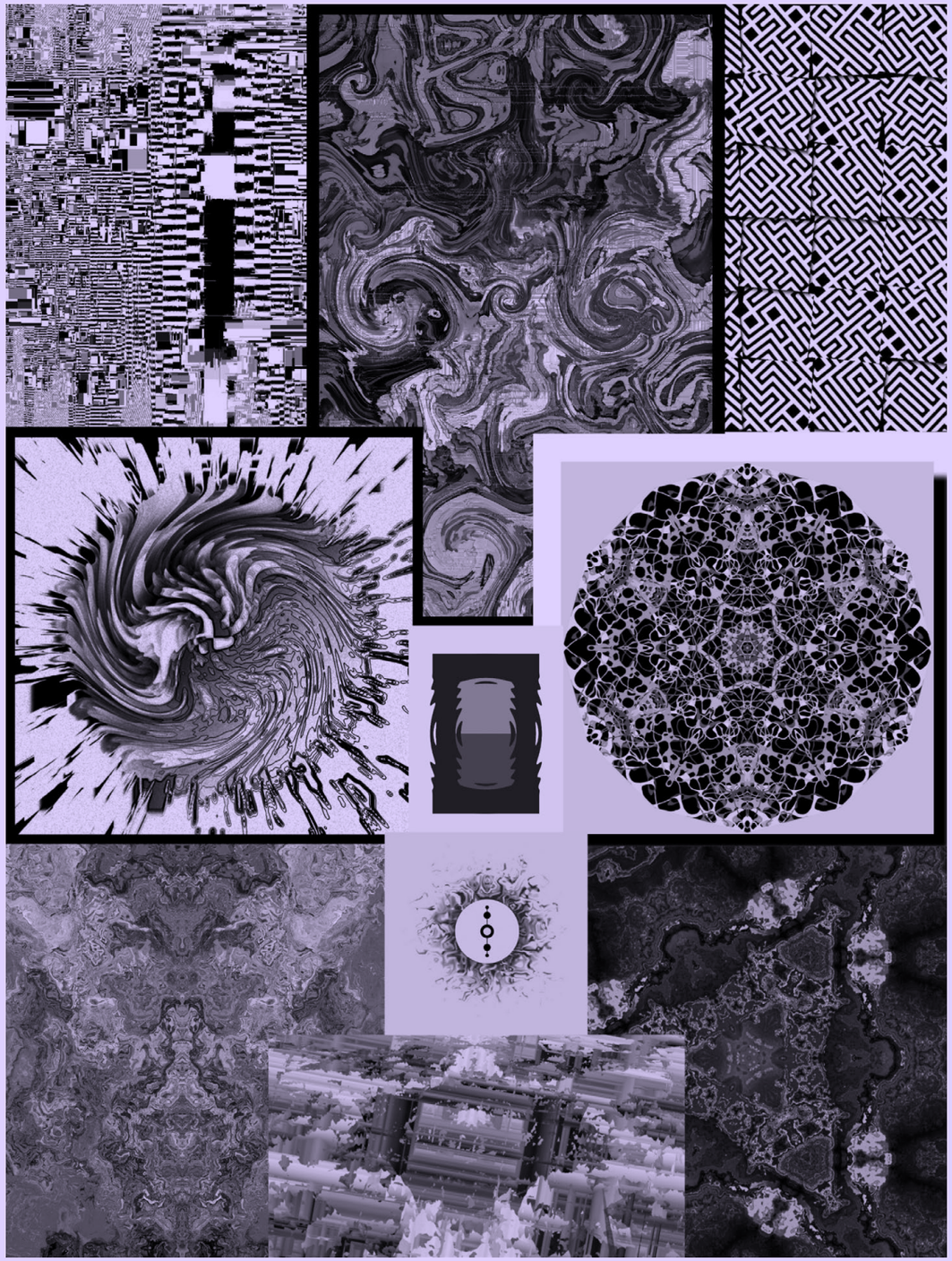


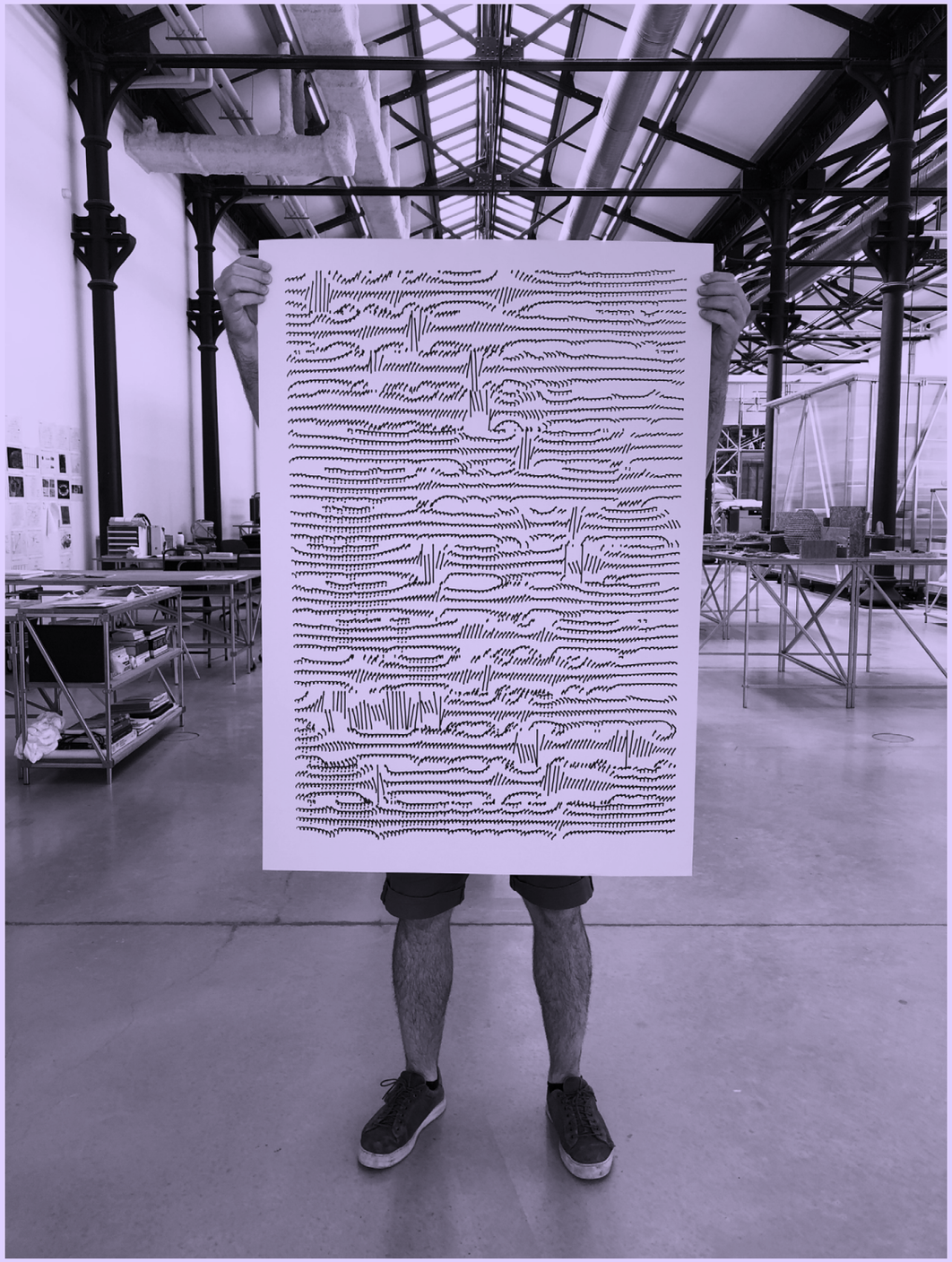



fluidtime project
ivrea, italy, 2004





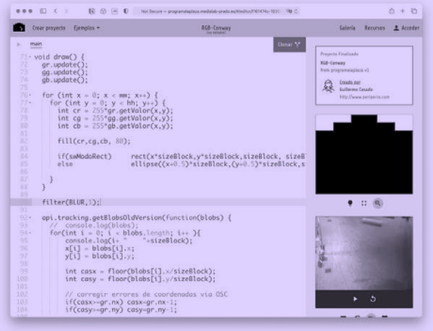




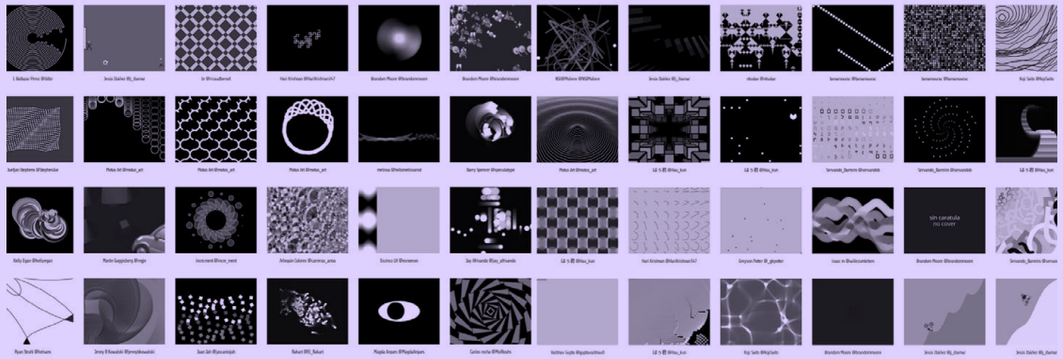


# ProgramaLaPlaza

Medialab-Prado, Madrid
2013-2020



En 2013 creamos una plataforma para que cualquier persona con conocimientos de processing pudiera programar y enviar sus sketches a  a la fachada digital de Medialab-Prado. Desde entonces y su cierre en 2021, hemos lanzado diversas "open-calls" en las que hemos recibido y proyectado cientos de sketches de todo el mundo.

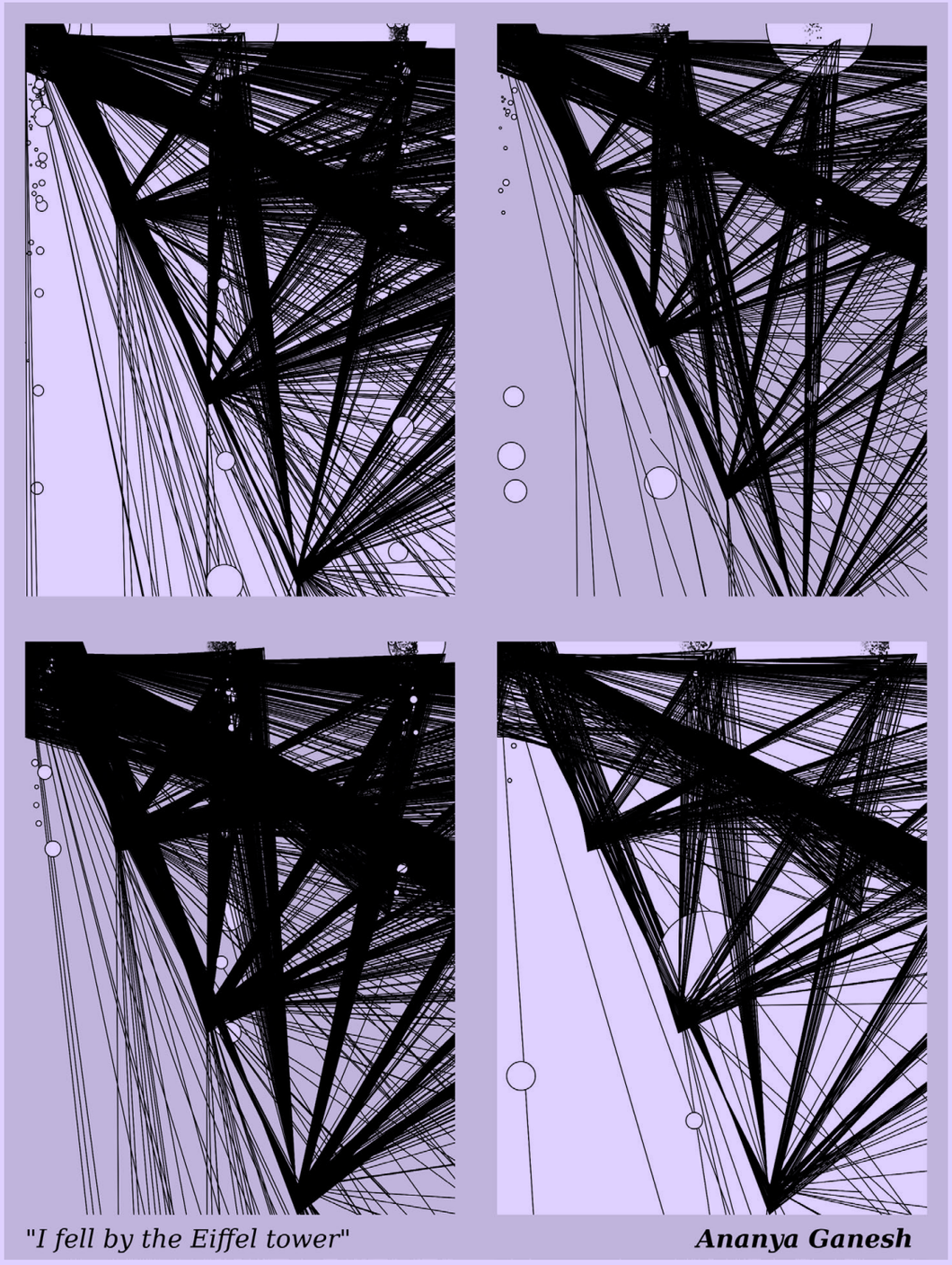





```
let shapes = [];

let line_gap,
    xoff, yoff, zoff,
    shapeXOff, shapeYOff,
    backgroundColour;

function setup() {
  createCanvas(windowWidth, windowHeight);
  noLoop();
  noFill();
  init();
}

function mousePressed() {
  init();
  draw();
}

function draw() {
  background(255);
  translate(width/2 + shapeXOff, height/2 + shapeYOff);

  for (let i = 0; i < shapes.length; i++) {

    let currentShape = [];

    beginShape();

    for (let j = 0; j < shapes[i].length; j++) {
      let x = shapes[i][j].x;
      let y = shapes[i][j].y;

      let angleNoise = 0;

      angleNoise = cos(sin(noise(sin(x * xoff) * PI, y * yoff, (x*xoff*y*yoff) + sin(frameCount*zoff) * TAU) * TAU) * TAU);

      curveVertex(
        x + map(angleNoise, 0, 1, -10, 10),
        y + map(angleNoise, 0, 1, -10, 10)
      );

      currentShape.push({
        x: x + map(angleNoise, 0, 1, -10, 10),
        y: y + map(angleNoise, 0, 1, -10, 10)
      });
    }

    curveVertex(currentShape[0].x, currentShape[0].y);
    curveVertex(currentShape[1].x, currentShape[1].y);
    curveVertex(currentShape[2].x, currentShape[2].y);
    endShape();
  }
}

function init() {
  resetMatrix();
  let weight = 2 + random((width+height) * 0.0001, (width+height) * 0.001);
  strokeWeight(weight);
  stroke(50);
  xoff = random(0.0001, 0.001);
  yoff = random(0.0001, 0.001);
  zoff = random(0.0001, 0.001);
  shapeXOff = random(-width/2, width/2);
  shapeYOff = random(-height/2, height/2);
  shapes = [];
  line_gap = random(2, 5);
  let size = random(15, 30);
  while (size <= width-width/5) {
    shapes.push(buildShape(size));
    size += line_gap + random(line_gap);
  }
}

function buildShape(size) {
  let shape = [];
  for (let i = 0; i <= TAU; i+=(5/size)) {
    shape.push({x: size * (sin(i) + size/shapeXOff*zoff), y: size * (cos(i) + size/shapeYOff)});
  }
  return shape;
}
```

derek_morash / gammastop

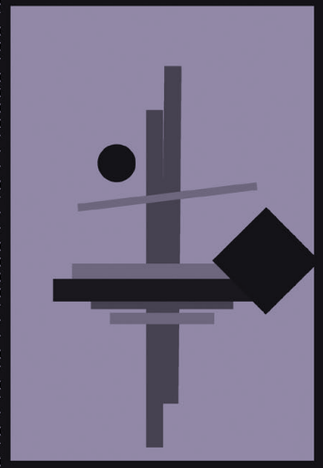
https://editor.p5js.org/derekmorash/sketches/K0nomgXVI



*"I fell by the Eiffel tower"* — ***Ananya Ganesh***



# Processing



Processing it all
The years of friendship
And creativity
The doors that opened
And the wind of opportunity that danced in

How lucky we are to process this moment

To breathe our fresh air of kindness

To push our boundaries of belief

To explore the poetry of code
And experience the brushstrokes of thought

The dilated pupils of understanding
And catalyzed curiosity

The sharing of stories

The sharing of techniques

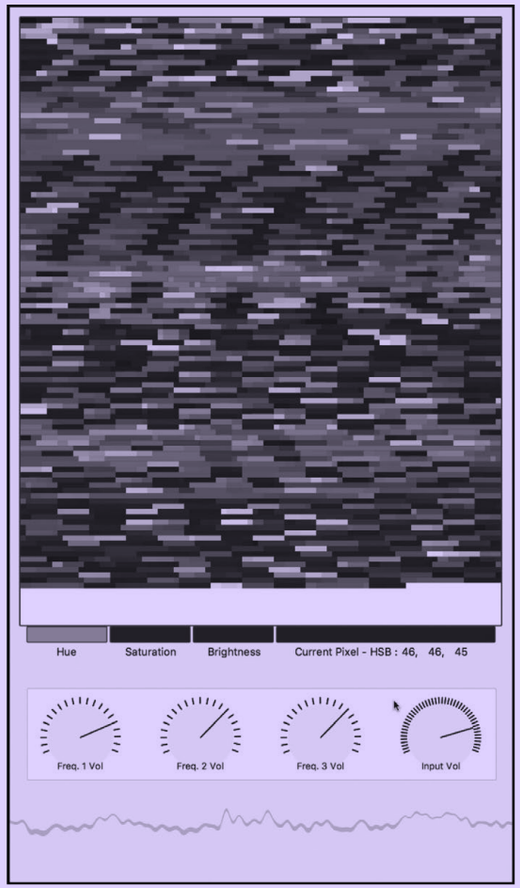
The critiques

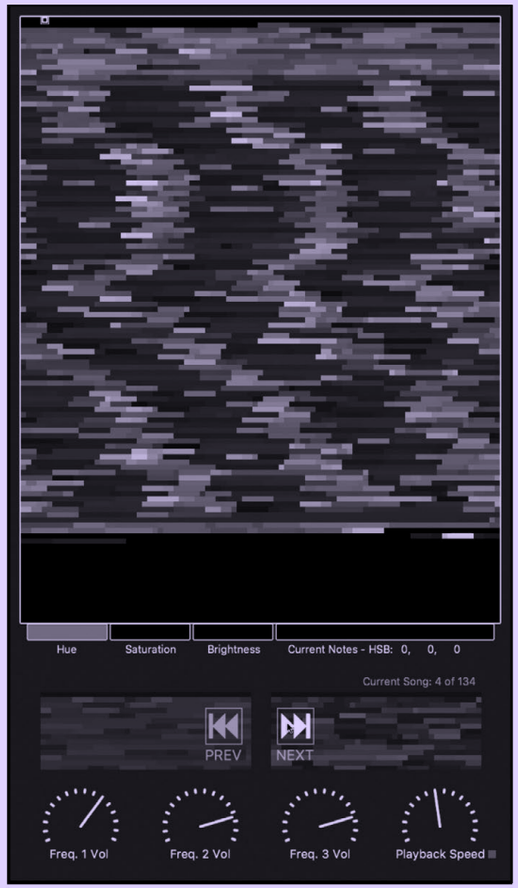
The love
The respect

To this community that's given me so much
Thank you.

Sofia Garcia





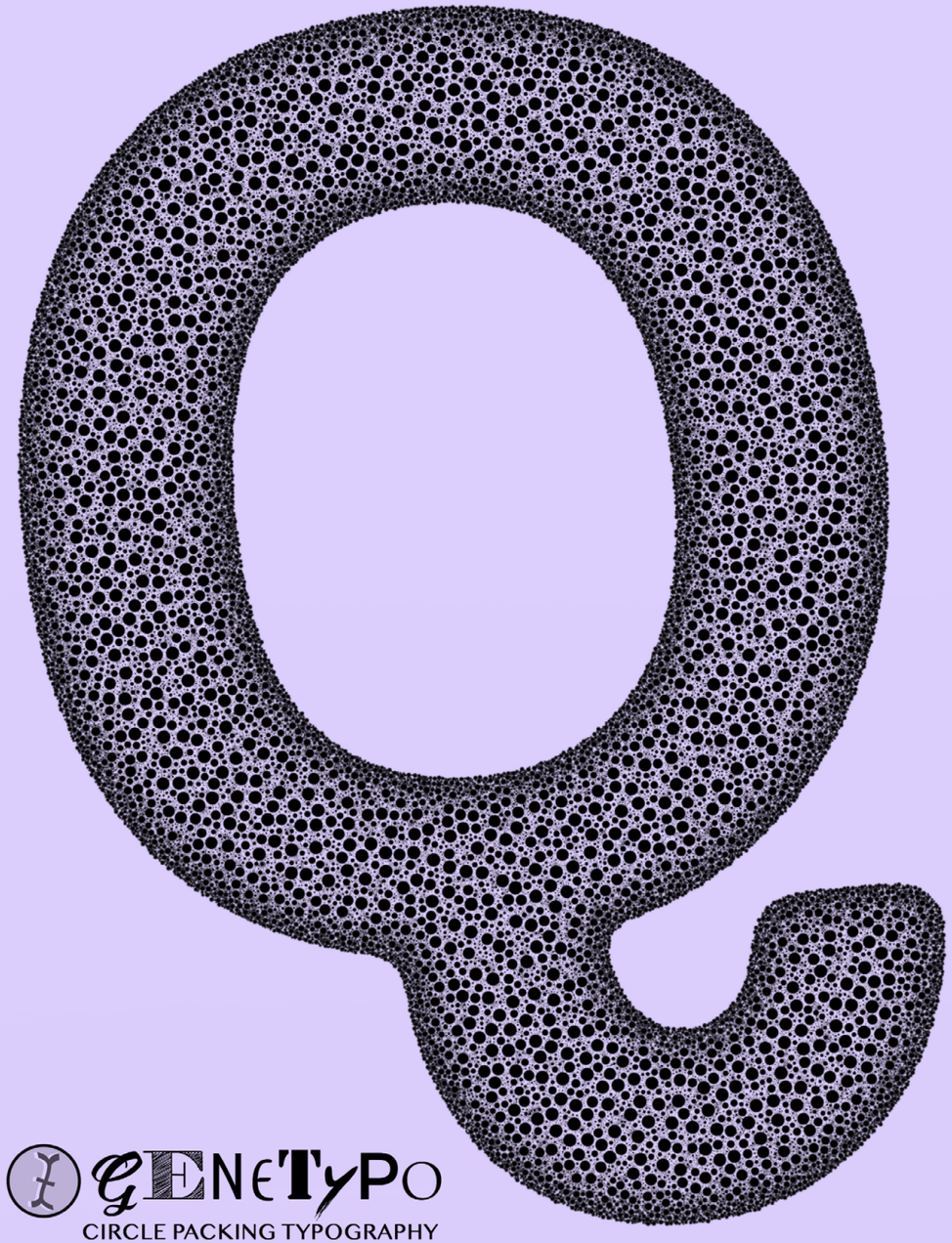

## Re: Perception

**Re: Perception** is a software artwork consisting of two applications. The first application generates images from audio input and the second generates audio output from pixel data stored in the resultant images.

The audio-to-image application (above left) analyses input audio to extract the three most dominant frequencies. Those three frequencies are stored as HSB color values that populate an image. Completed images are saved as png files.

The image-to-audio application (above right) plays HSB color values from png files as three separate sine wave frequencies. Controls to advance to next "audio-image" as well as change sine wave volumes and playback speed are included.

**@garrettbeleu**

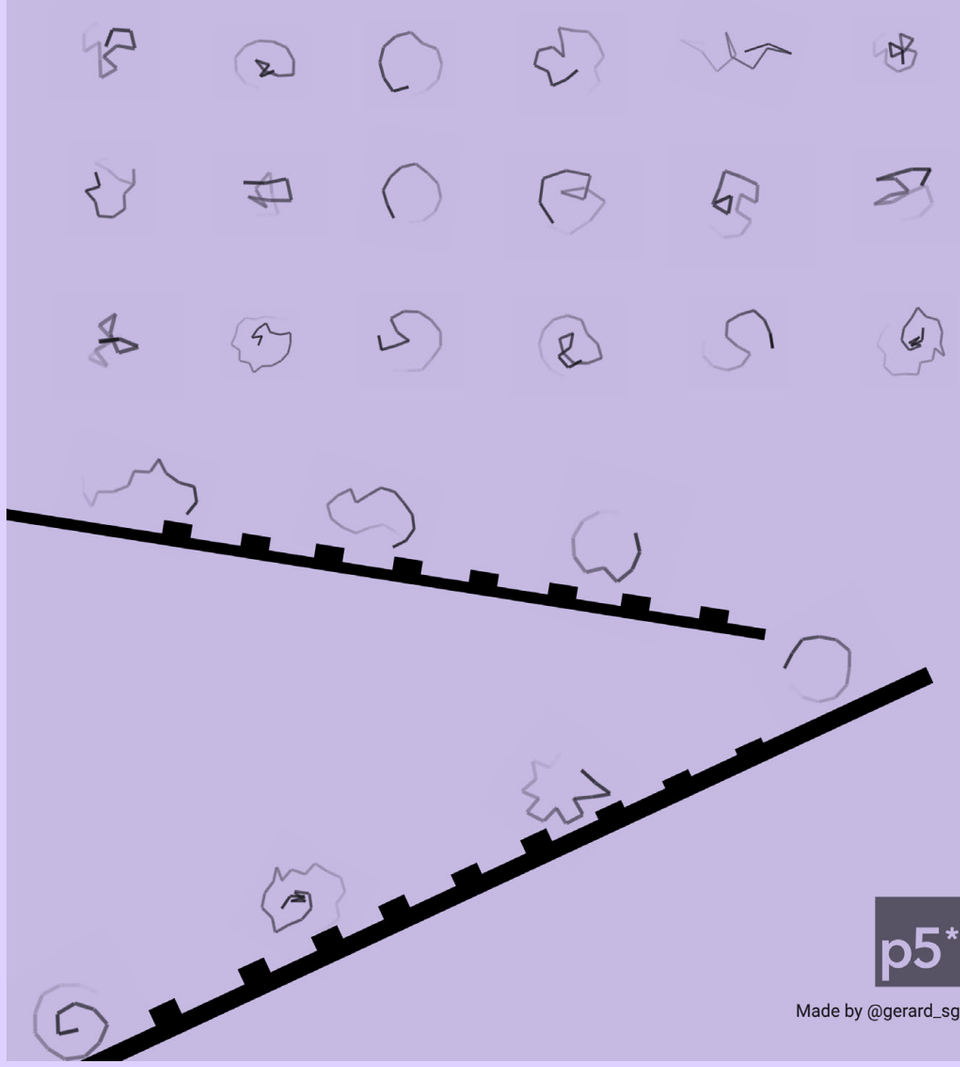


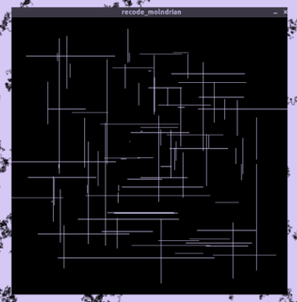


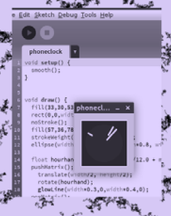

# COEVO

A 2D physically-based environment with artificial agents that create 2D solutions for given problems

"Create an object that moves through an inclined plane"





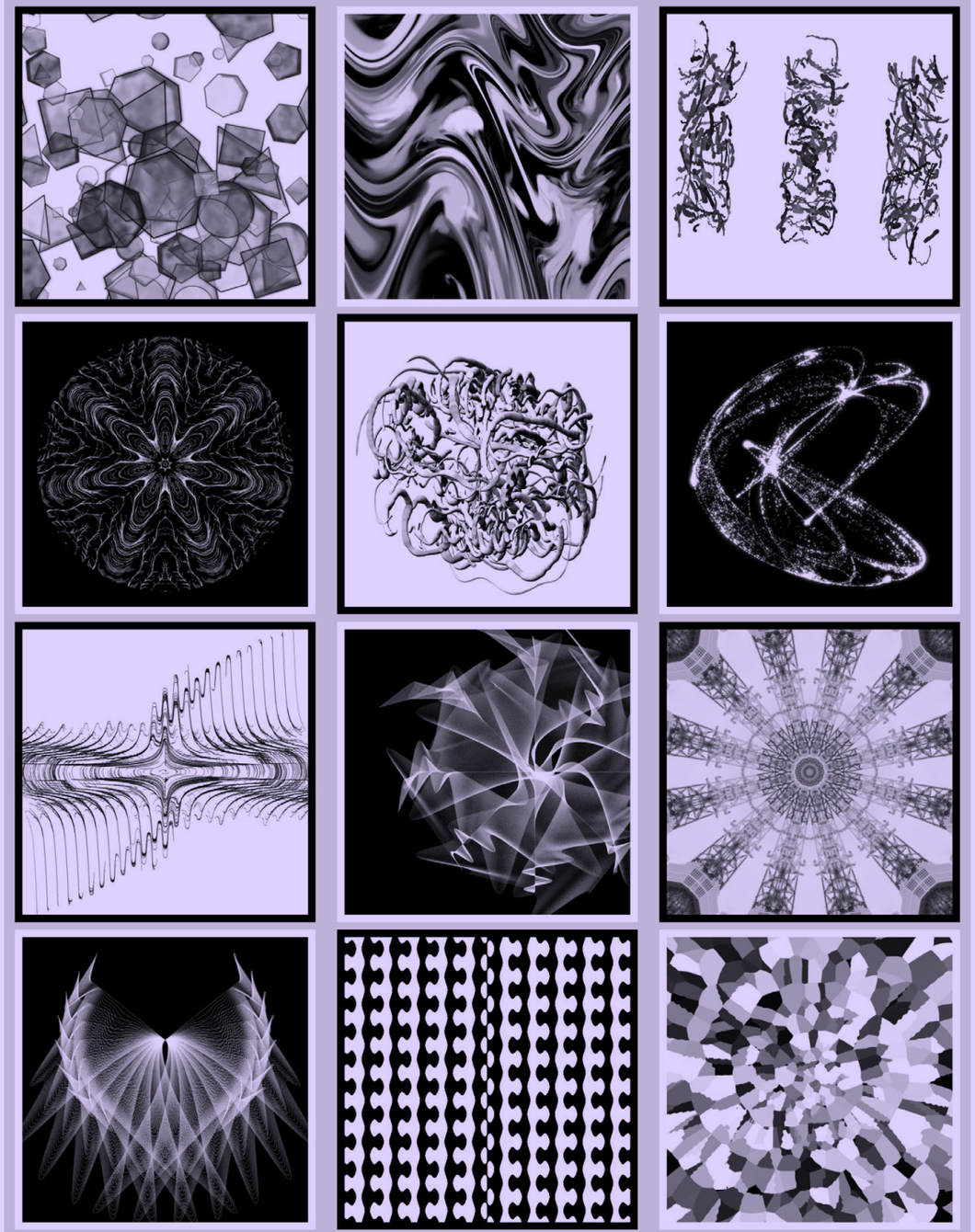

thanks to the Processing community for the amazing 20 years of sharing, building, contributing, and creating. it's been an honor to be able to play with Processing and the myriad of libraries and offshoots, participate in local community groups, and experiment with everything from designing necklaces to making a creative app to just generally messing around with patterns, chaos, and geometry.

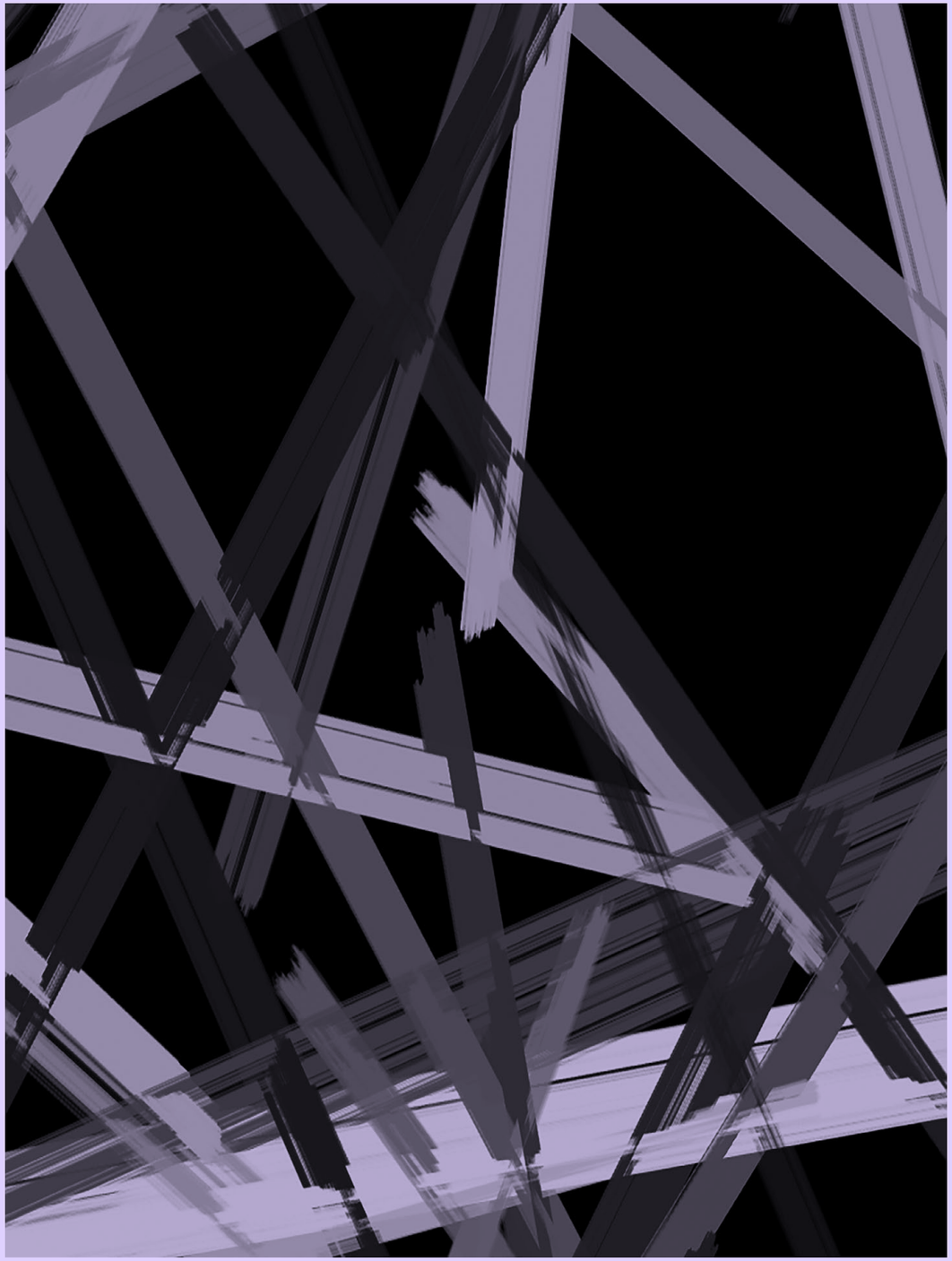
- josh giesbrecht



## ギン / GIN

twitter, instagram : @gin_graphic







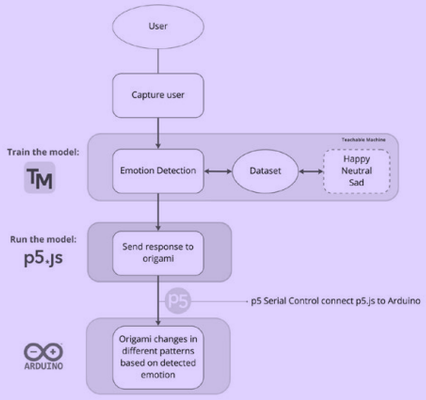
# ORIFLECT

**Imagine if the architecture can change shape based on people experiences...**

ORIFLECT is an interactive structure with a facial emotion recognition camera that changes its color as well as the pattern to reflect the viewer's expression.

Form Study: Pleat Tessellation Origami

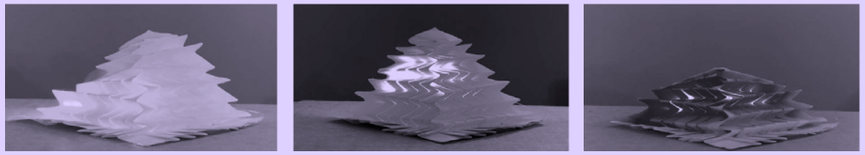
## Project Work Flow




## Design Process


https://www.youtube.com/watch?v=ipua7CqM14o

## Final Design



**Happy face detected**

The origami will dance, and the color change to blinking green.

**Natural face detected**

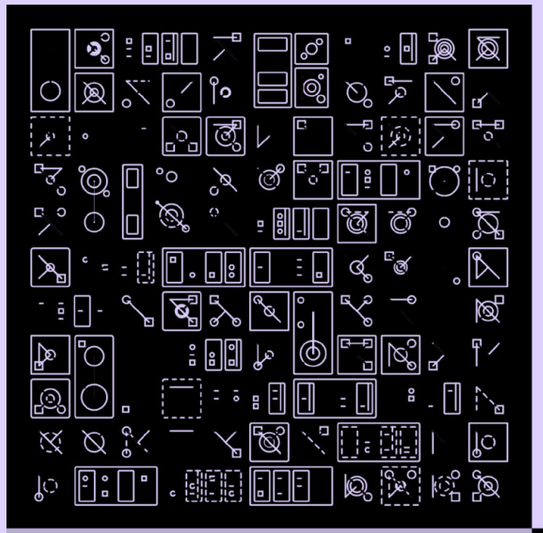
The origami will come to its original state, and the color change to white.

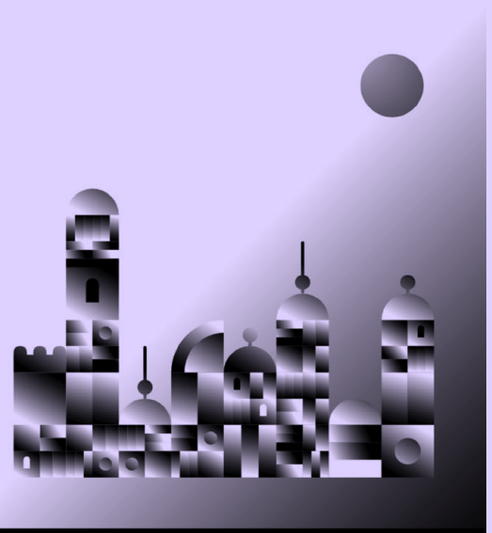
**Sad face detected**

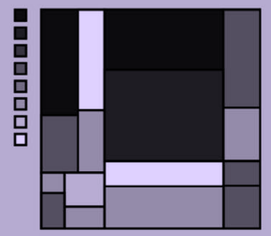
The origami will close, and the color change to purple.

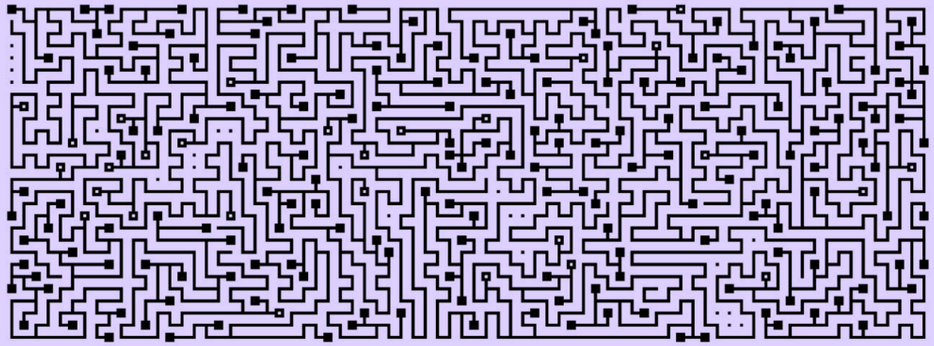
**Link to project video:** https://www.youtube.com/watch?v=S8GY1EHmXPE

Zahrasadat Golestanha



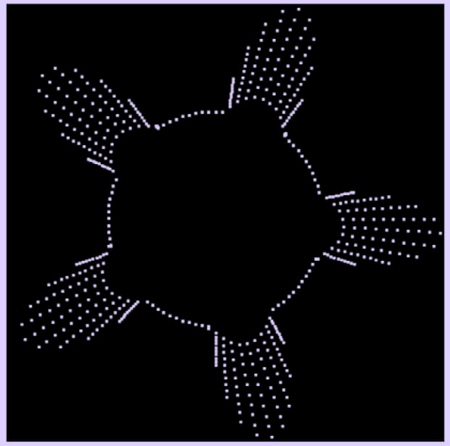


**Gorilla Sun – Ahmad Moussa**
Made with processing, p5 and love <3
To many more years of creative coding!

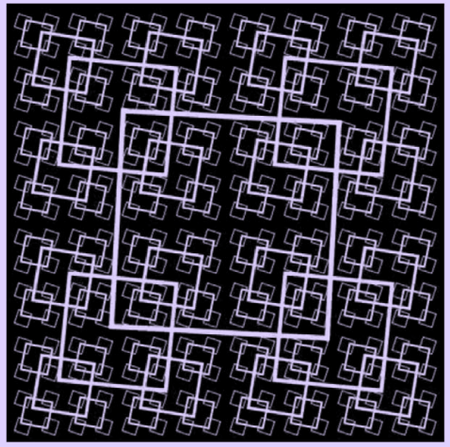
# Movement

I like making art with Processing. I think the most appealing aspect of creating arts with Processing is the ability to bring movement to the art. I'm not good at creating beautiful designs, but I can and do like to create interesting movements.



Even if it doesn't use complicated shapes, I can keep looking at it as long as the movement is interesting. I would like to continue looking for movements that I find interesting.

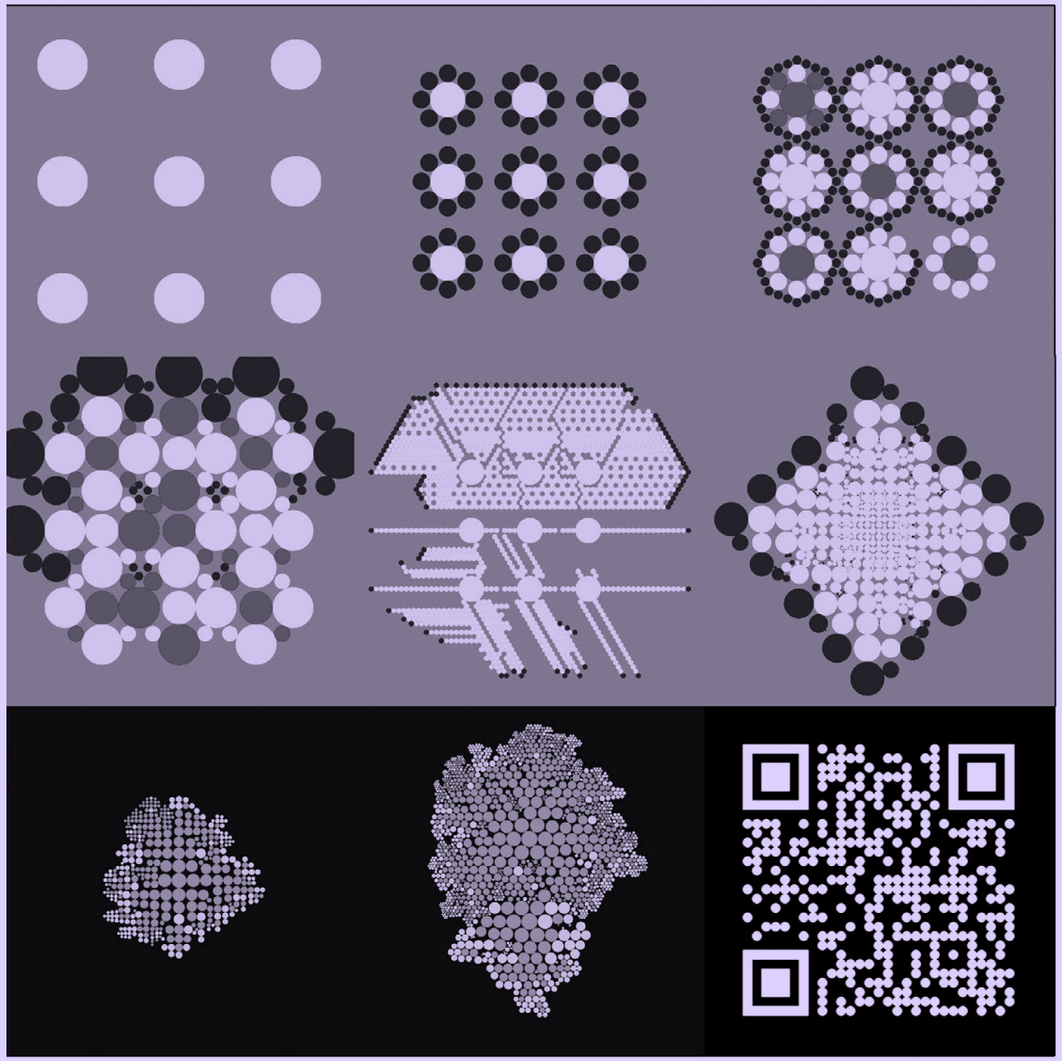
My arts are in the following repository!
https://github.com/gotutiyan/generative-art-Processing



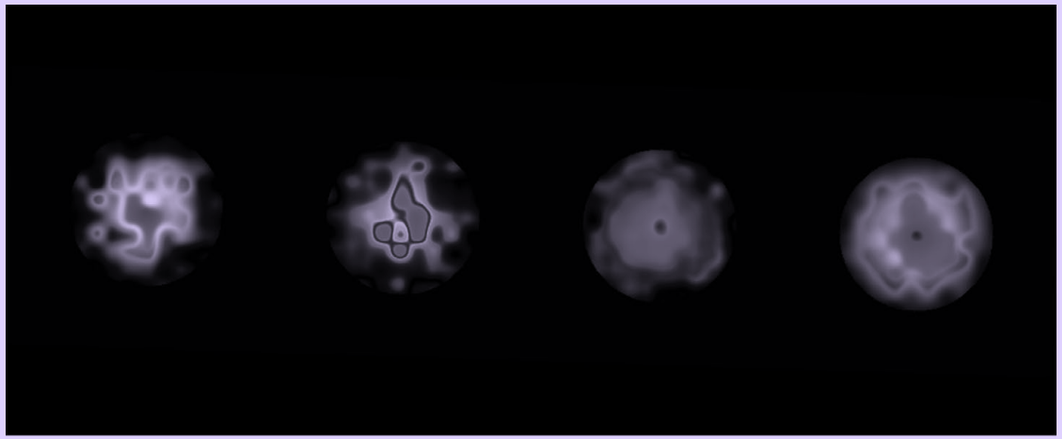


# Processing is Experimentation and Iteration

Various parts of a generative design need to be developped when implementing a new idea.
The output often reveals unexpected visual results which inspire to experiment further.
**Capturing a frame each time the code runs** during development gives a unique insight in the algotrithm and design decisions.



Result



```
// Alien Blob 2.1 (Feb 15 '06)
// based on formula by Jim Bumgardner
// implementation by Ryan Govostes

float xoff = 0, yoff = 0, zoff = 0;
float sineTable[];
float dTable[];

void setup() {
  size(256, 256);
  colorMode(HSB, 2);
  background(0);
  noiseDetail(1);
  // framerate(24);

  // precalculate 1 period of the sine wave (360 degrees)
  sineTable = new float[360];
  for (int i = 0; i < 360; i ++) sineTable[i] = sin(radians(i));

  dTable = new float[width*height];
  float  cx = width/2;
  float  cy = height/2;
  int n = 0;
  for (int y = 0; y < height; y++) {
    for (int x = 0; x < width; x++) {
      if (x == cx && y == cy) {
        dTable[n] = 0;
      }
      else {
        // dTable[n] = log(dist(x,y,cx,cy)); // another fun one...
        dTable[n] = dist(x,y,cx,cy) * .05;
      }
      ++n;
    }
  }
}
```

```
void draw() {
  float d, h, s, b, n;
  float xx;
  float yy = yoff;
  int w2 = width / 2;
  int h2 = height / 2;
  int offset = 0;

  int m = millis();
  float varH = sineTable[(m>>3)%360];
  float varS = sineTable[(m>>4)%360];
  float varN = sineTable[(m>>5)%360];

  loadPixels();

  for (int y = 0; y < height; y ++) {
    xx = xoff;
    for (int x = 0; x < width; x ++) {
      d = dTable[offset];
      if (d <= 3.5) {
        n = noise(xx, yy, zoff)*(1+varN*.5); // noise only needs to
                                             // be computed once per pixel

        // use pre-calculated sine results
        h = 1 + sineTable[int(degrees(d + n *(3+varH*2))) % 360];
        s = sineTable[int(degrees(d + n * (5+varS*4))) % 360] + 1;
        b = sineTable[int(degrees(d + n * 3)) % 360] + 1;

        // determine pixel color
        pixels[offset] = color(h, s, b);
      }
      offset++;
      // increment xx
      xx += 0.0625; // = x/16.0
    }

    // this value only needs to be changed once per pixel line
    yy += 0.0625;
  }

  // move through noise space -> animation
  xoff += 0.3;
  yoff += 0.07;
  zoff += 0.1;

  updatePixels();
}
```

ONE OF MY FIRST SKETCHES, IN 2003, WAS A PORT OF AN EFFECT FROM THE 1992 MAC APPLICATION "PIXEL MAGIC" BY JIM BUMGARDNER. THIS VERSION IS THE RESULT OF A COLLABORATION BETWEEN JIM AND I TO IMPROVE THE SPEED. I LEARNED TRICKS LIKE BUILDING LOOKUP TABLES FOR TRIG FUNCTIONS.

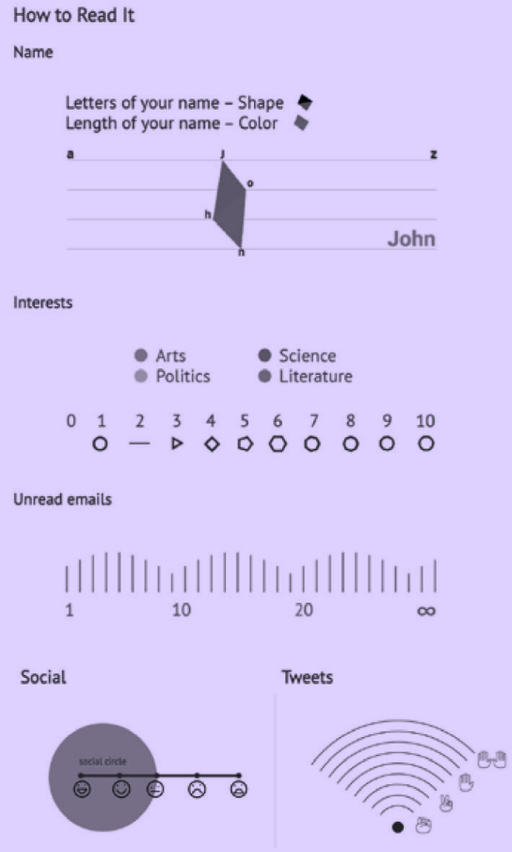


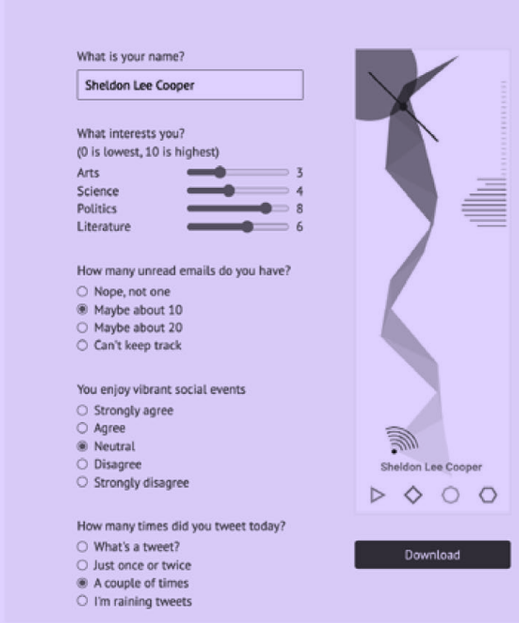
# Data Imprint

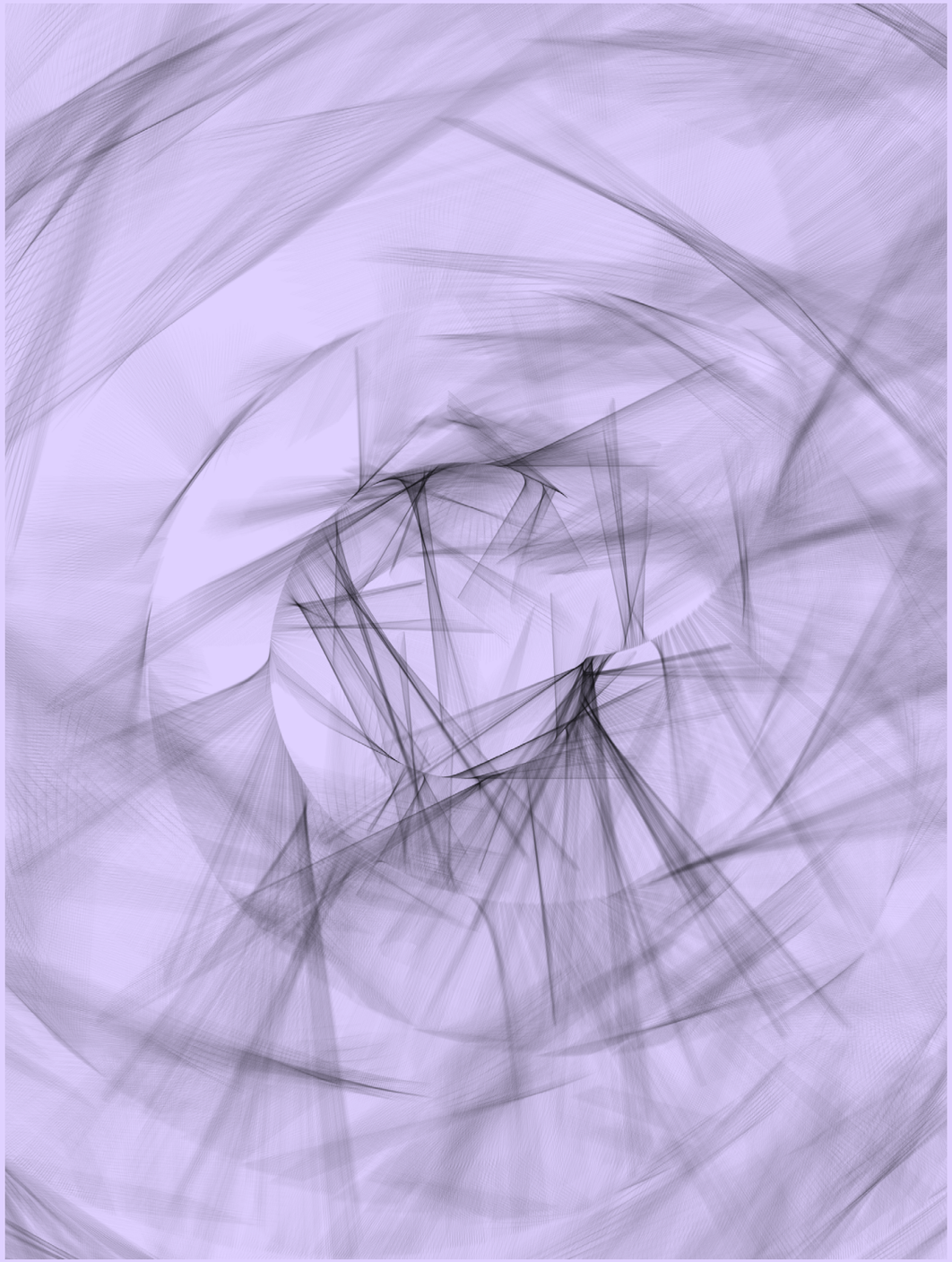
**Gramener**
gramener.com/dataimprint

Everyone's different: from our names and habits, to our social preferences, we set ourselves apart as unique bundles of traits.

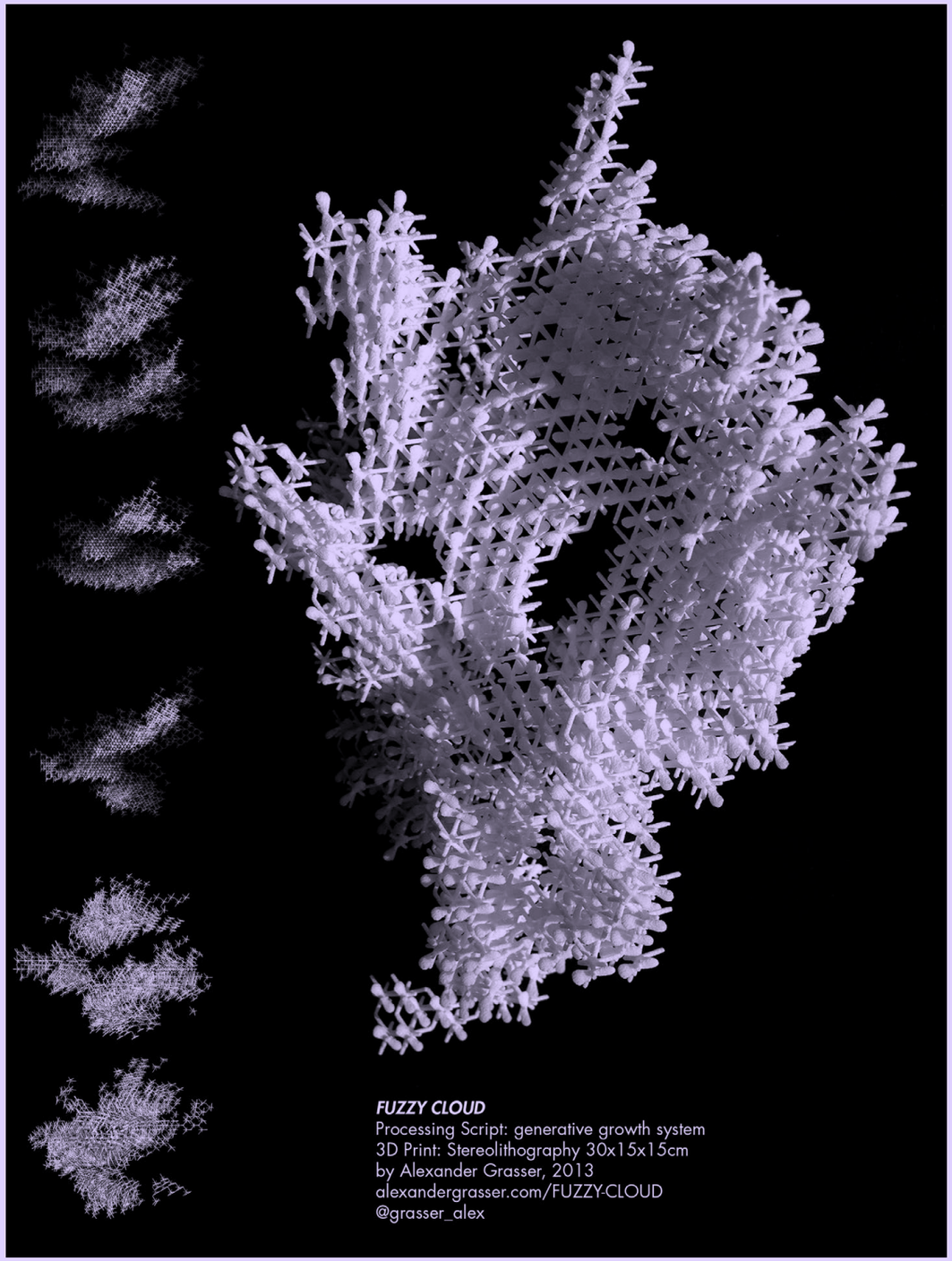
This generative data art project aims at capturing the distinct portrait of an individual. Great for badges, bookmarks and mementos!






FUZZY CLOUD
Processing Script: generative growth system
3D Print: Stereolithography 30x15x15cm
by Alexander Grasser, 2013
alexandergrasser.com/FUZZY-CLOUD
@grasser_alex



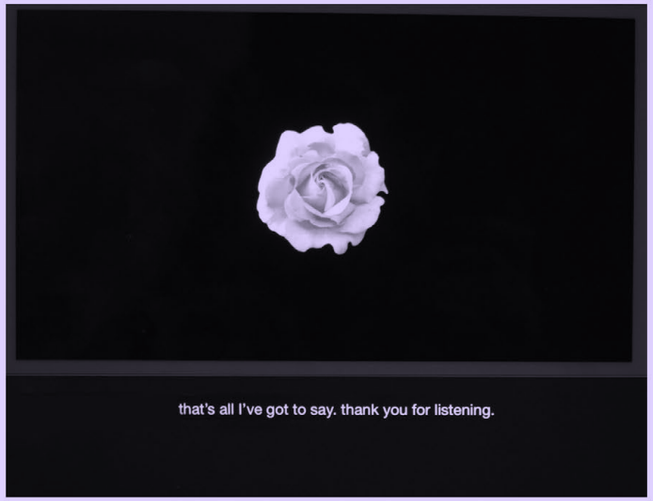




*image from 1.0 release speech by lauren lee mccarthy*

# The Sketch Metaphor

The Sketch Metaphor

p5* 2020 v1.0

*These trees are magnificent,*
*but even more magnificent is the sublime*
*and moving space between them,*
*as though with their growth it too increased.*

*Rainer Maria Rilke*

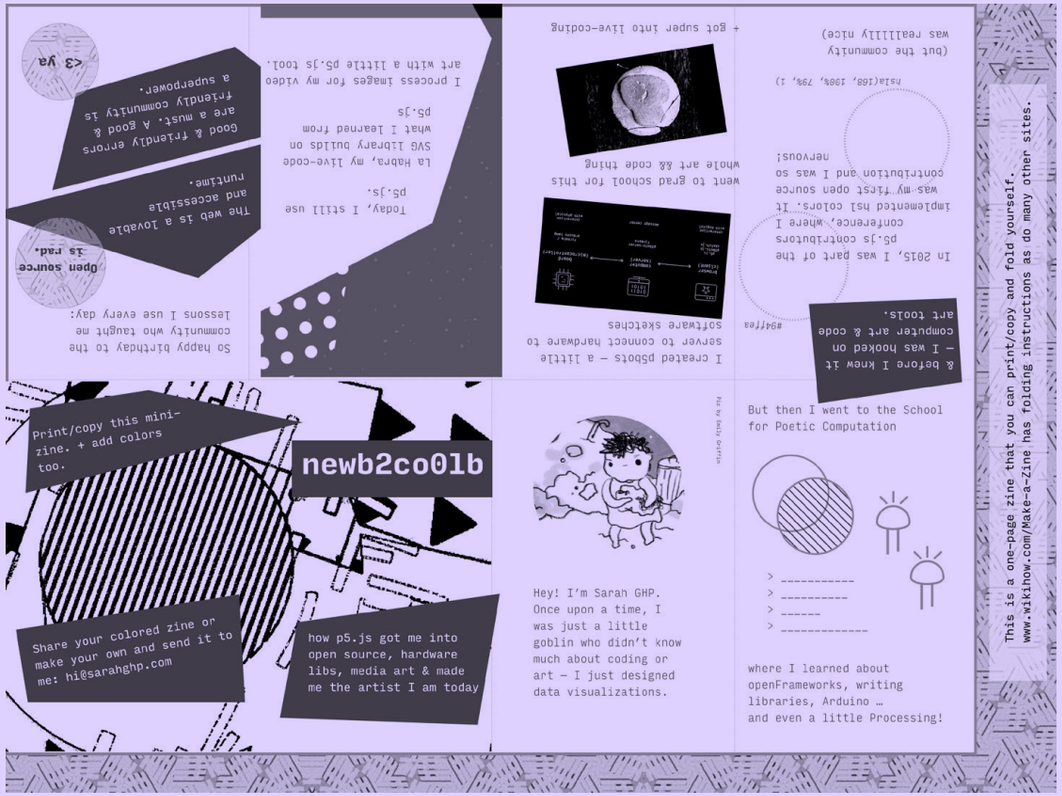
There is a warm and knowable expansiveness in the act of drawing. Those two axes, that blankness, are universal. Drawing is the first inconceivably complex possibility space that many of us encounter. Despite this, there is something definitively accessible about drawing.

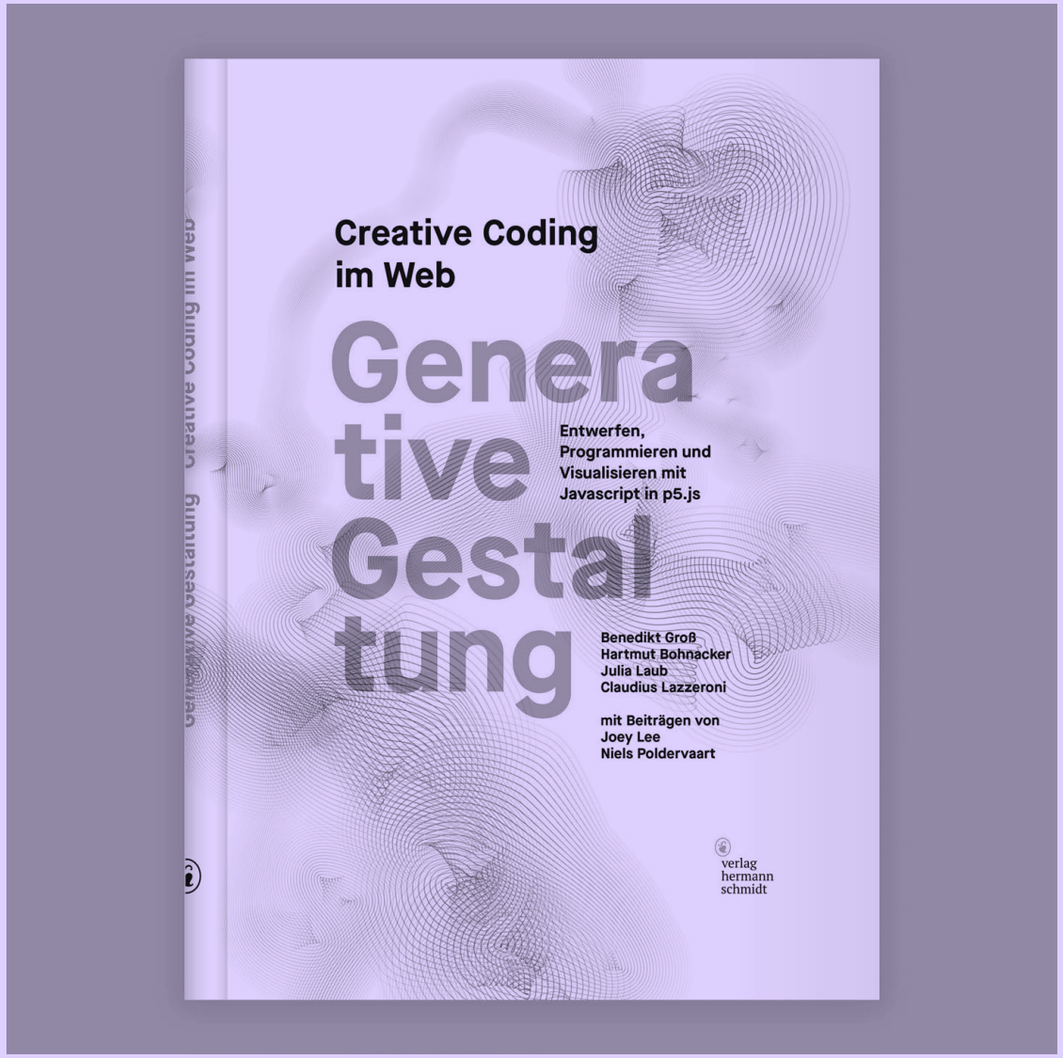
Drawing never feels exclusive, never insists that anyone should not explore or experiment. The boundlessness of drawing is a commonly-held good. The vastness of the act is inviting, not alienating. It feels like anyone can find a new way to speak color, to excavate texture, to defy composure, to suggest a path for the eye. The tilting and gliding of hand across surface, the trailing mark, these are not secret knowledge.

Code is a line etched into the possible, a route for matter that has been charged. Coding lives both inside and outside of social and political life. Unfortunately, so often it carries the baggage of supposed expertise. Gatekeepers insist that coding belongs to a specialized-class; that it is historically inherited or fundamentally inaccessible. But we know that they aren't right and this move is meant to hide something ancient and mischievous. For this too is a canvas, a contiguous shape; a stage for procedural play between ourselves and others.

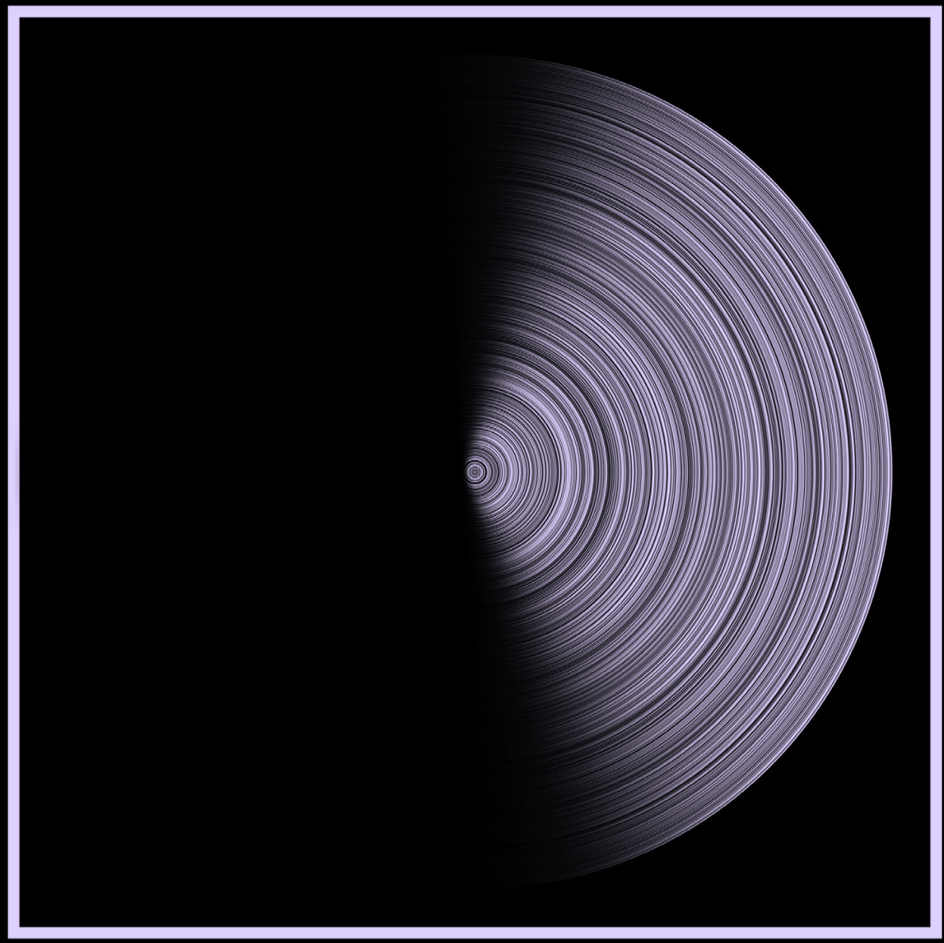
This show brings together an archive that demonstrates code as a medium of exploration, of learning, of vulnerability, of curiosity. These are snapshots of artists *feeling it out*. They are windows into an uncomplicated delight.

*- Stalgia Grigg 2.25.2020*


newb2co01b
Print/copy this mini-zine. + add colors too.
Share your colored zine or make your own and send it to me: hi@sarahghp.com
how p5.js got me into open source, hardware libs, media art & made me the artist I am today
Hey! I'm Sarah GHP. Once upon a time, I was just a little goblin who didn't know much about coding or art — I just designed data visualizations.
But then I went to the School for Poetic Computation
where I learned about openFrameworks, writing libraries, Arduino … and even a little Processing!
This is a one-page zine that you can print/copy and fold yourself.
www.wikihow.com/Make-a-Zine has folding instructions as do many other sites.
I created p5bots — a little server to connect hardware to software sketches
went to grad school for this whole art && code thing
+ got super into live-coding
(but the community was realllly nice)
In 2015, I was part of the p5.js contributors conference, where I implemented hsl colors. It was my first open source contribution and I was so nervous!
& before I knew it — I was hooked on computer art & code art tools.
Today, I still use p5.js.
la Habra, my live-code SVG library builds on what I Learned from p5.js
I process images for my video art with a little p5.js tool.
Good & friendly errors are a must. A good & friendly community is a superpower.
<3 ya
The web is a lovable and accessible runtime.
Open source is rad.
So happy birthday to the community who taught me lessons I use every day:


Creative Coding im Web
Generative Gestaltung
Entwerfen, Programmieren und Visualisieren mit Javascript in p5.js
Benedikt Groß
Hartmut Bohnacker
Julia Laub
Claudius Lazzeroni
mit Beiträgen von
Joey Lee
Niels Poldervaart
verlag hermann schmidt



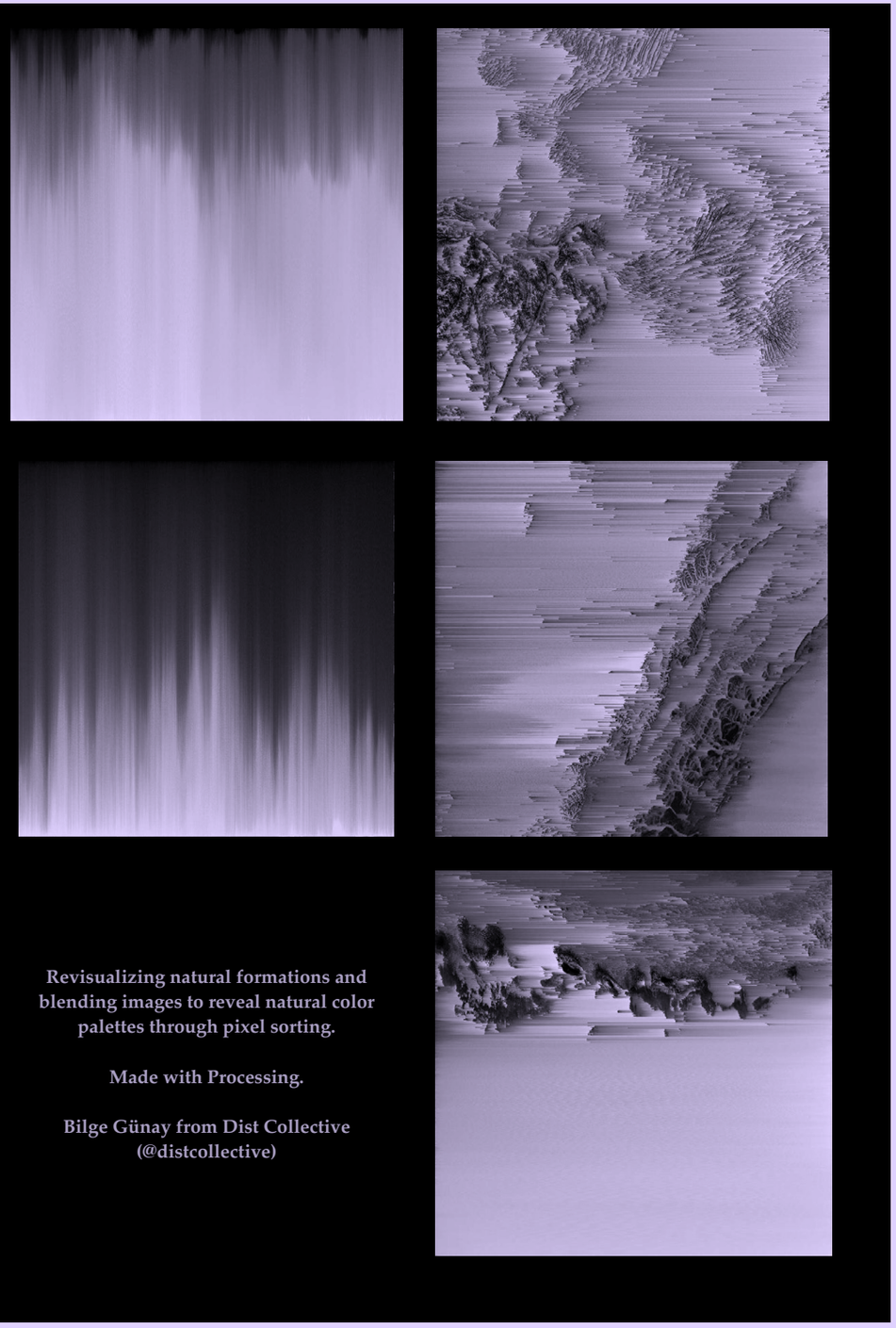


## Ops #1

[2400x2400] - 700 circles in grayscale.
"14021982" was used as random seed.
Generative image created with code (p5.js, javascript/canvas).

NFT minted: 2021-05-12 — editions: 10
By Nuno Guedes Silva

http://nuno.ws/nft

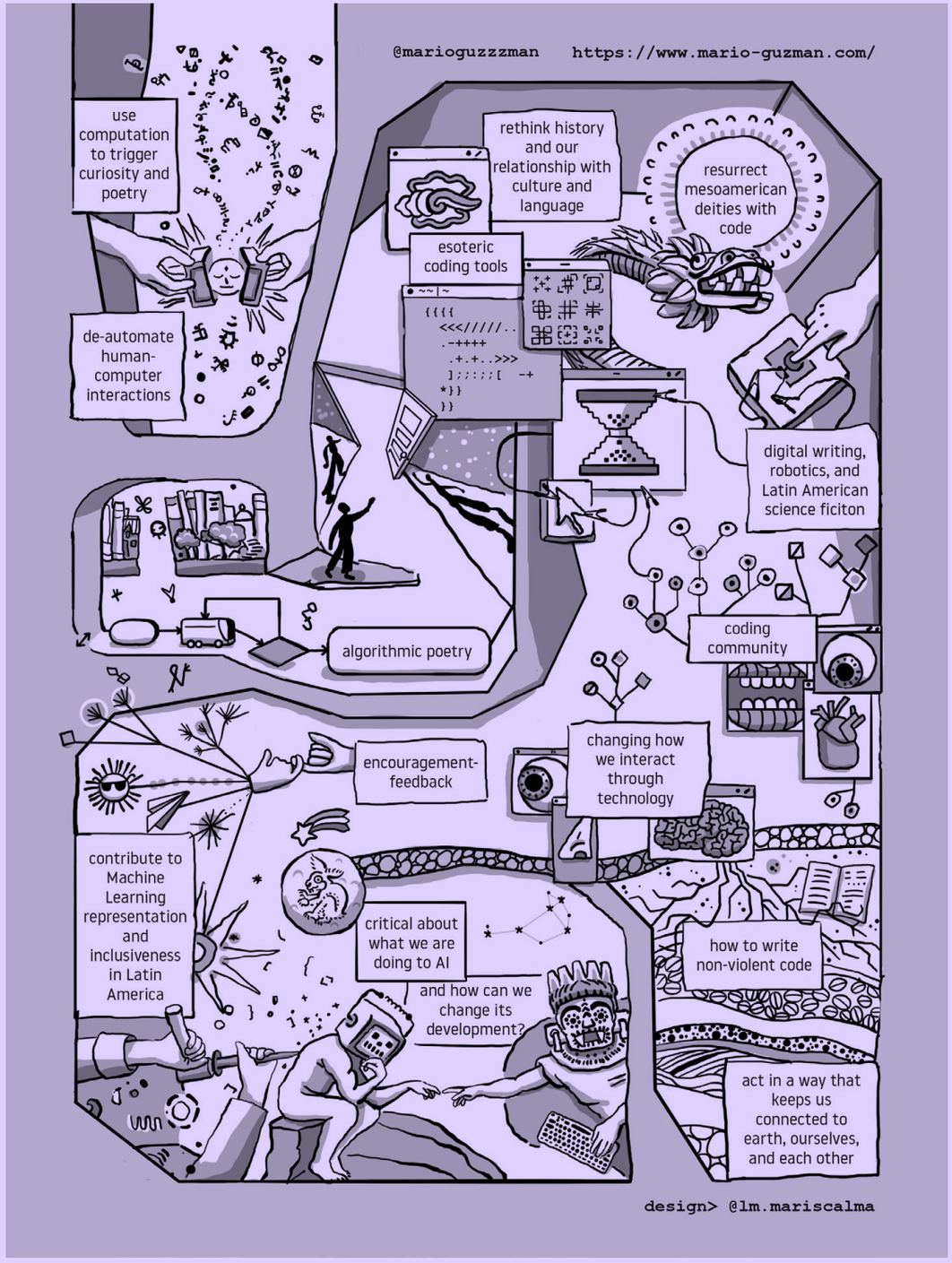


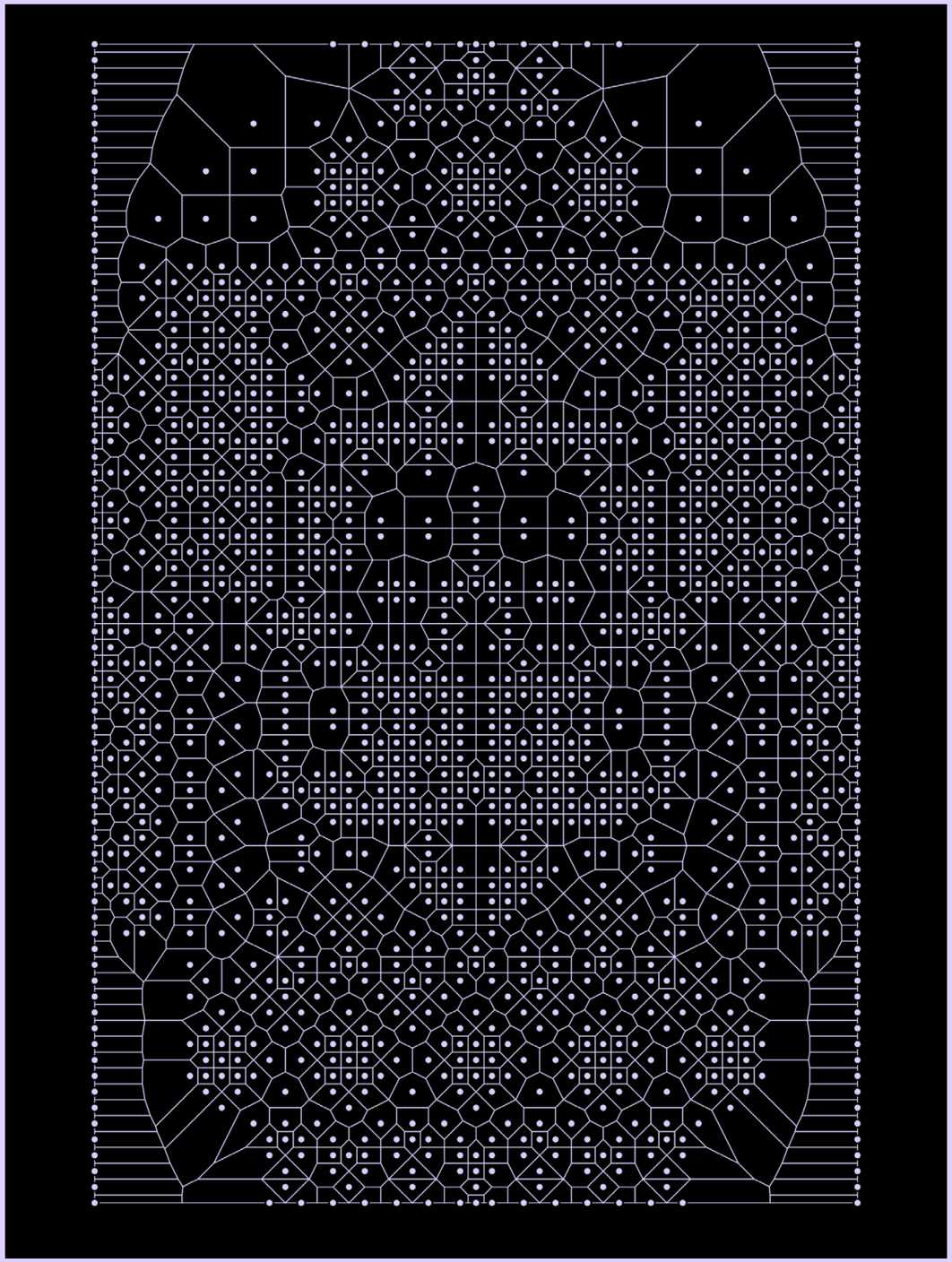
Revisualizing natural formations and blending images to reveal natural color palettes through pixel sorting.

Made with Processing.

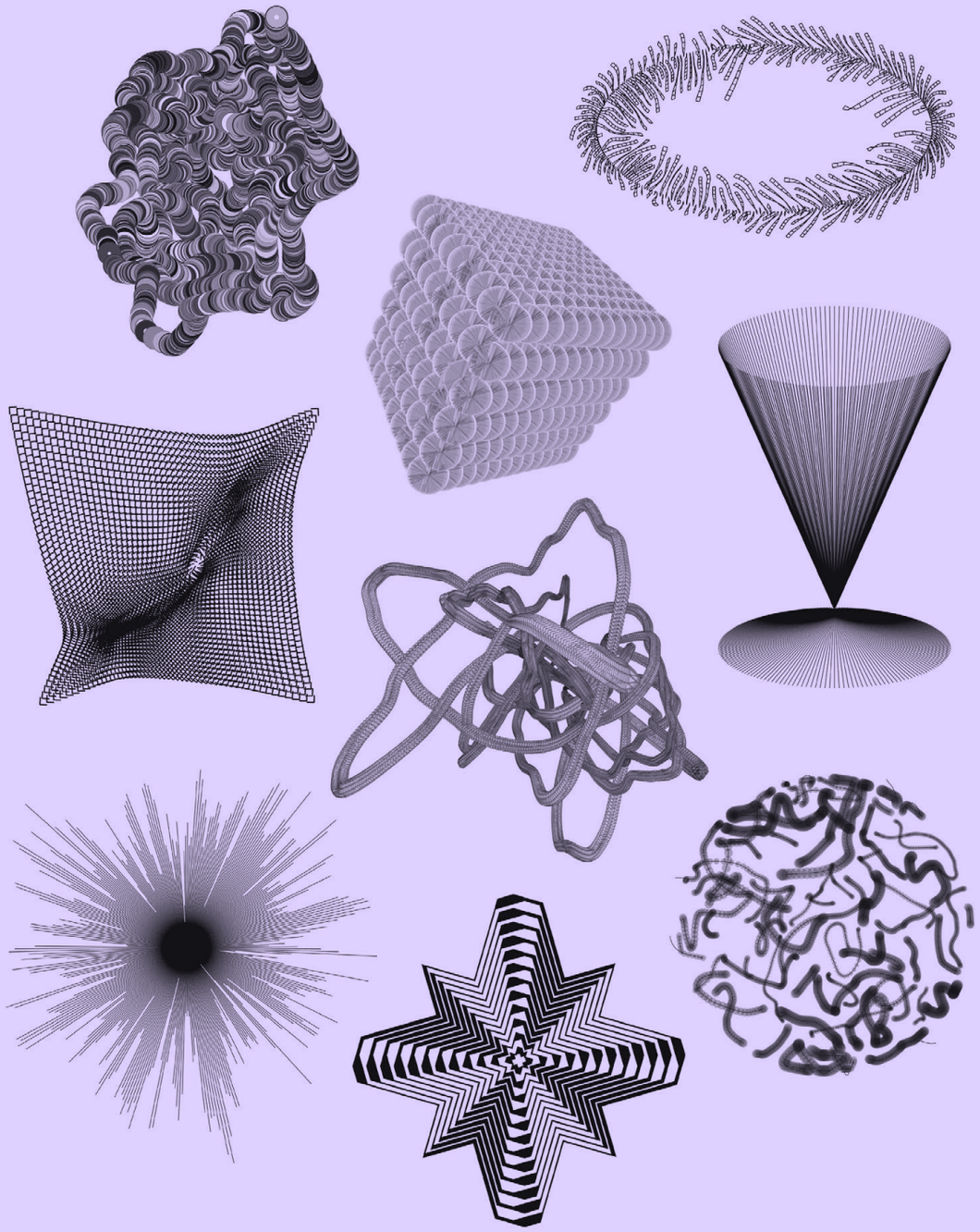
Bilge Günay from Dist Collective (@distcollective)

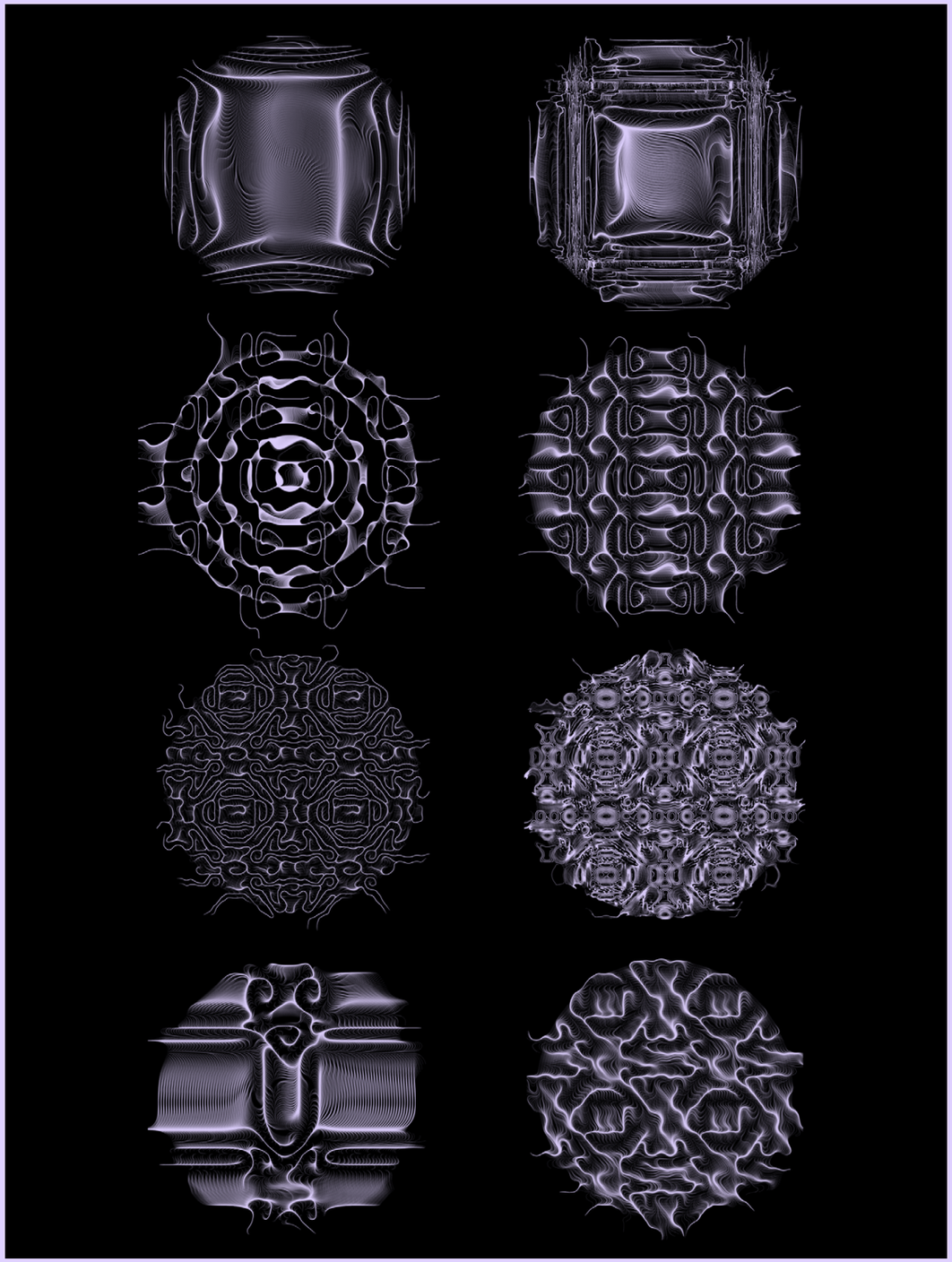




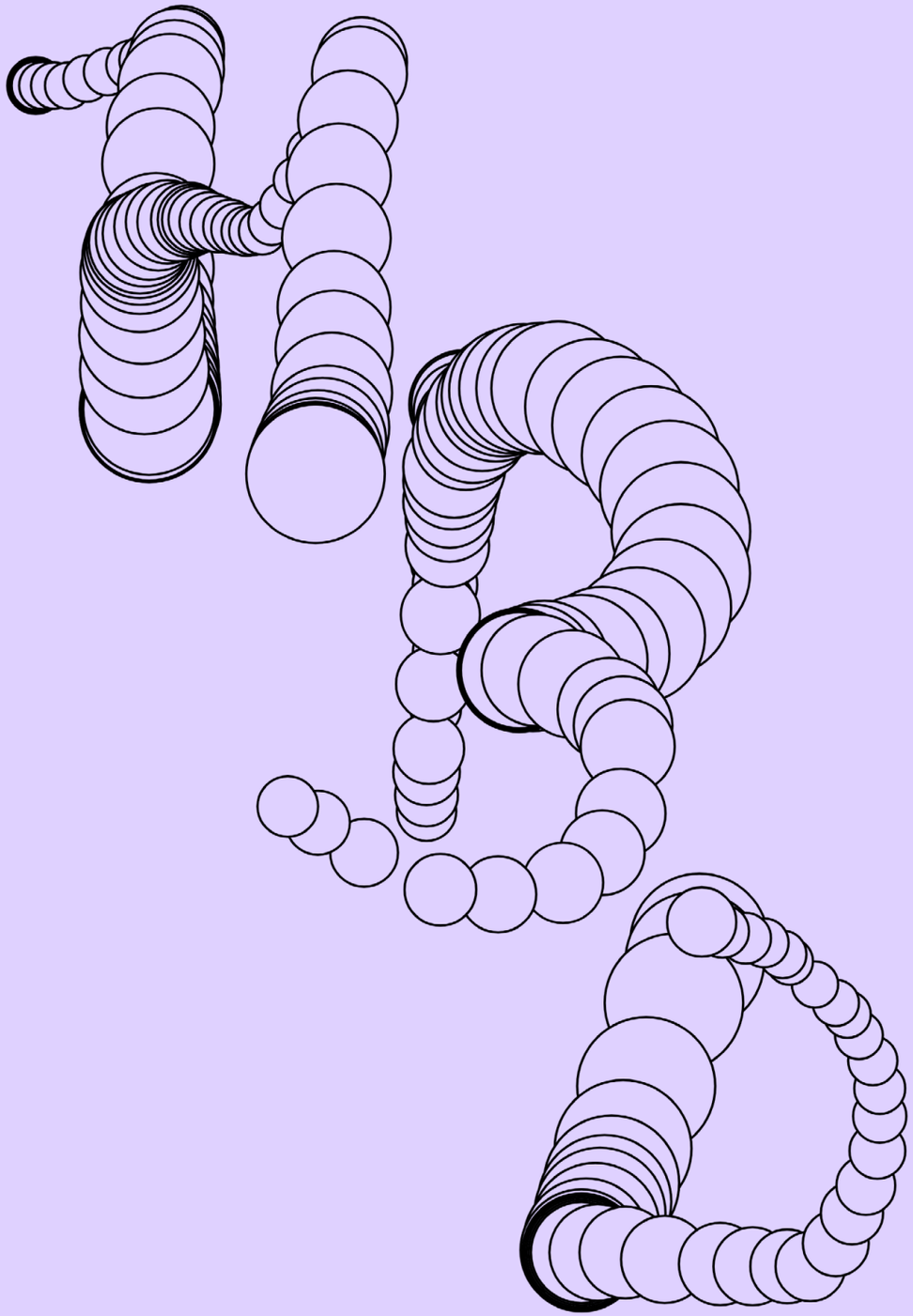




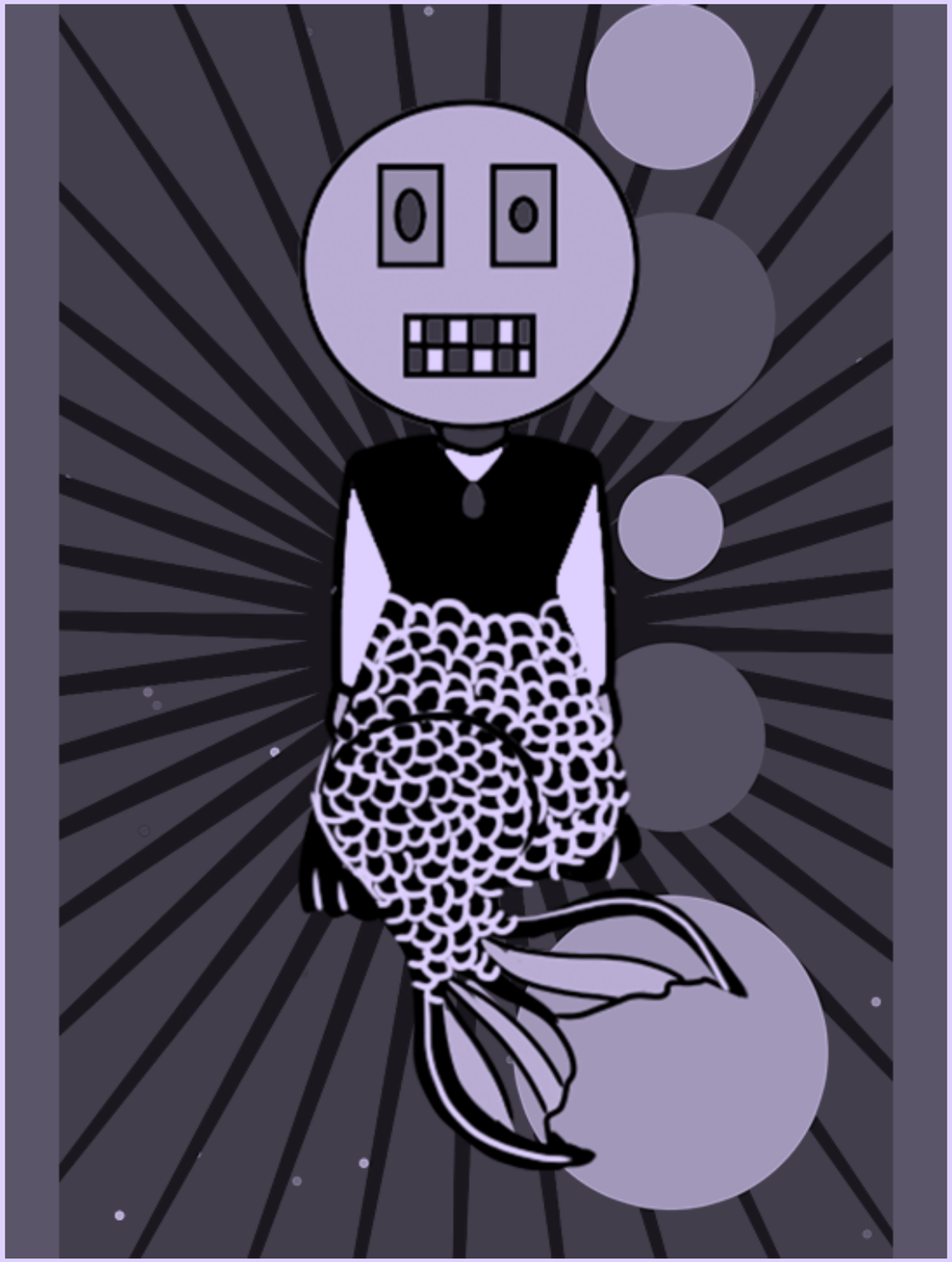




HANIF HAGHTALAB , NOISE FIELD STUDY CON 211 526



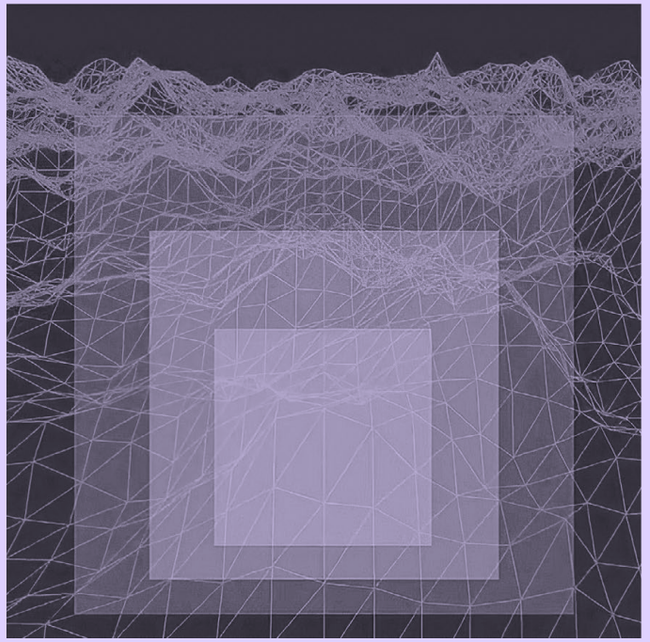
GOTTFRIED HAIDER CON 212 527





## Planting a Tree

A kneeling woman, planting a tree. Redrawn from the motive on the German 50 cent coin from before the EURO.



## Homage to Josef Albers‘ “Homage to the Square“

3D terrain generation with Perlin Noise behind semi-transparent squares.

I first heard about Processing (or what was to become Processing) in 2001, when Casey Reas told me about it. We were both doing a residency in Italy and I must admit that, while I applauded the idea of expanding access to coding tools and making it quicker and easier to get useful results, I didn't think it had much relevance to my own work in interactive architecture and environments. It wasn't until a couple of years later, when I asked Casey to come and talk about it to my students in Unit 14 (a.k.a. the Interactive Architecture Workshop at the Bartlett School of Architecture) and he showed us an astonishing variety of things that people had created with it – in particular controlling things in the physical world! – that I really began to understand the potential it offered. Apart from all the other stuff (which was obvious to others, but had taken me a while to grasp) he showed how it collapsed the distinctions between concept / sketch / prototype / experiment / tool / input / output / finished work – and this is exactly what I'd been trying to do in my own interactive architecture work.

Once Arduino appeared on the scene (and Firmata! Oh Firmata!!) the deal was sealed, and I ended up using Processing in almost every project I did for 15 years, even if it didn't drive the 'final' thing. I used it to control and manage sensors and actuators; to explore what was eventually to become Pachube; to design interfaces for responsive physical environments; to create an energy monitoring app; to drive cathedral façade and building scale projections and even massive water screens on beaches. I used it to build email processing scripts, and data visualisations once Pachube was dealing with tens of millions of data points per day, and to analyse real-time audio from an acoustically coupled neural network. I've used it to simulate coral growth, to design a massive urban laser event, to give voices to mussels underwater, and even to control the visuals in an interactive pedestrian crossing. Through it all, the P5 community has been an essential part of the process – learning from others, always learning, and, occasionally helping others (I hope…) and witnessing the same joy emerging in their faces as they experience a new paradigm of creativity.

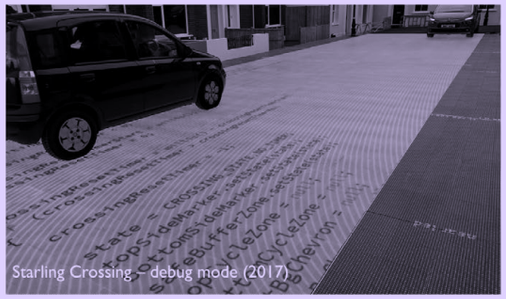
But one of my fondest memories of Processing is from Christmas Day 2019, when Ling and I just wanted some gentle peace and quiet. So we sat on our bed, gently coding in the Processing IDE, side-by-side. Marital bliss.

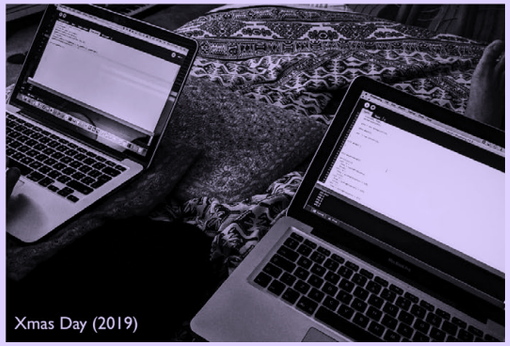
Thank you Casey, Ben, for kicking it all off and thank you to the thousands of others that have made it flourish.

– Usman Haque • haque.co.uk

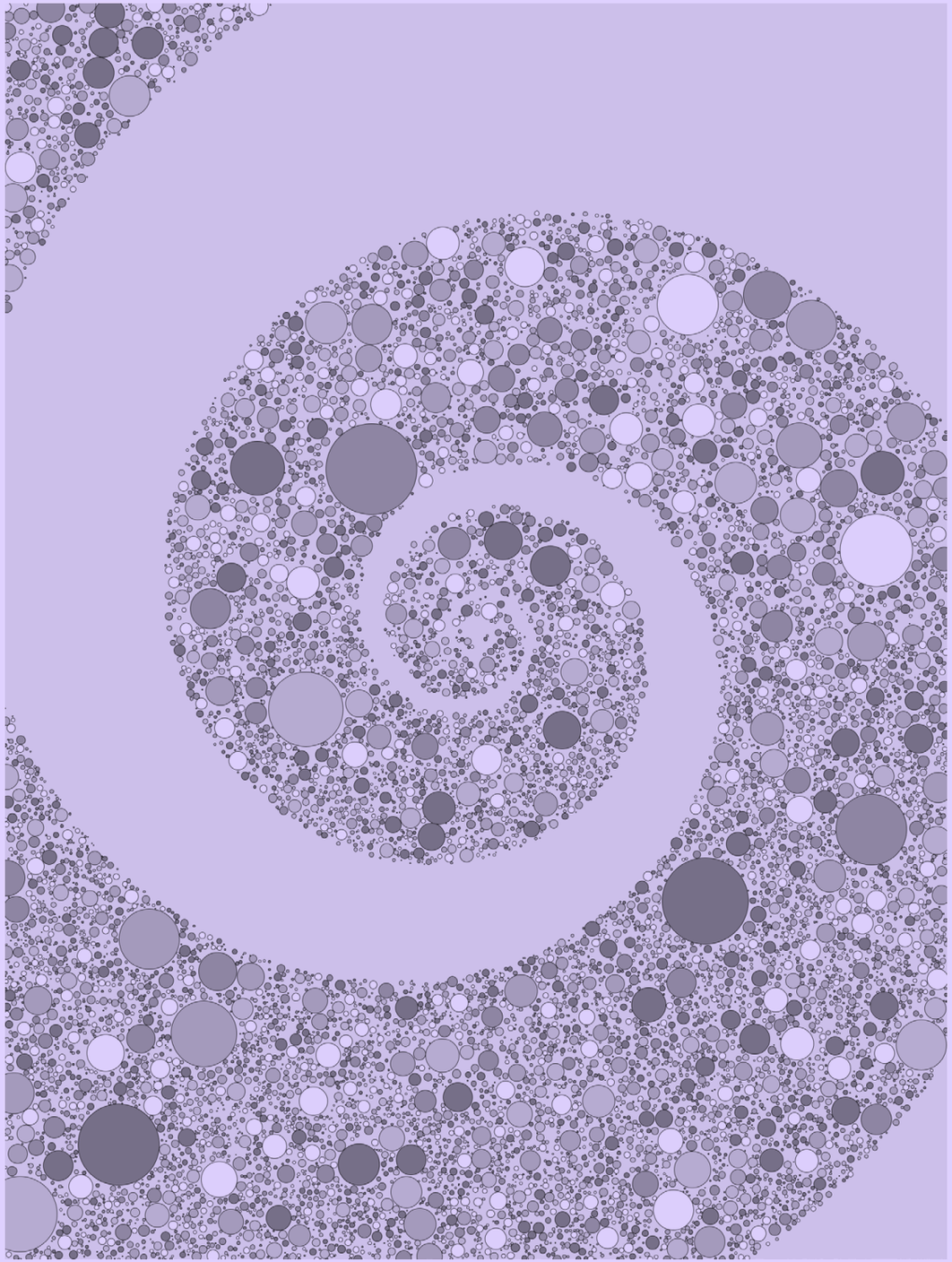

Primal Source (2007)


Evoke (2008)


Starling Crossing – debug mode (2017)


Xmas Day (2019)







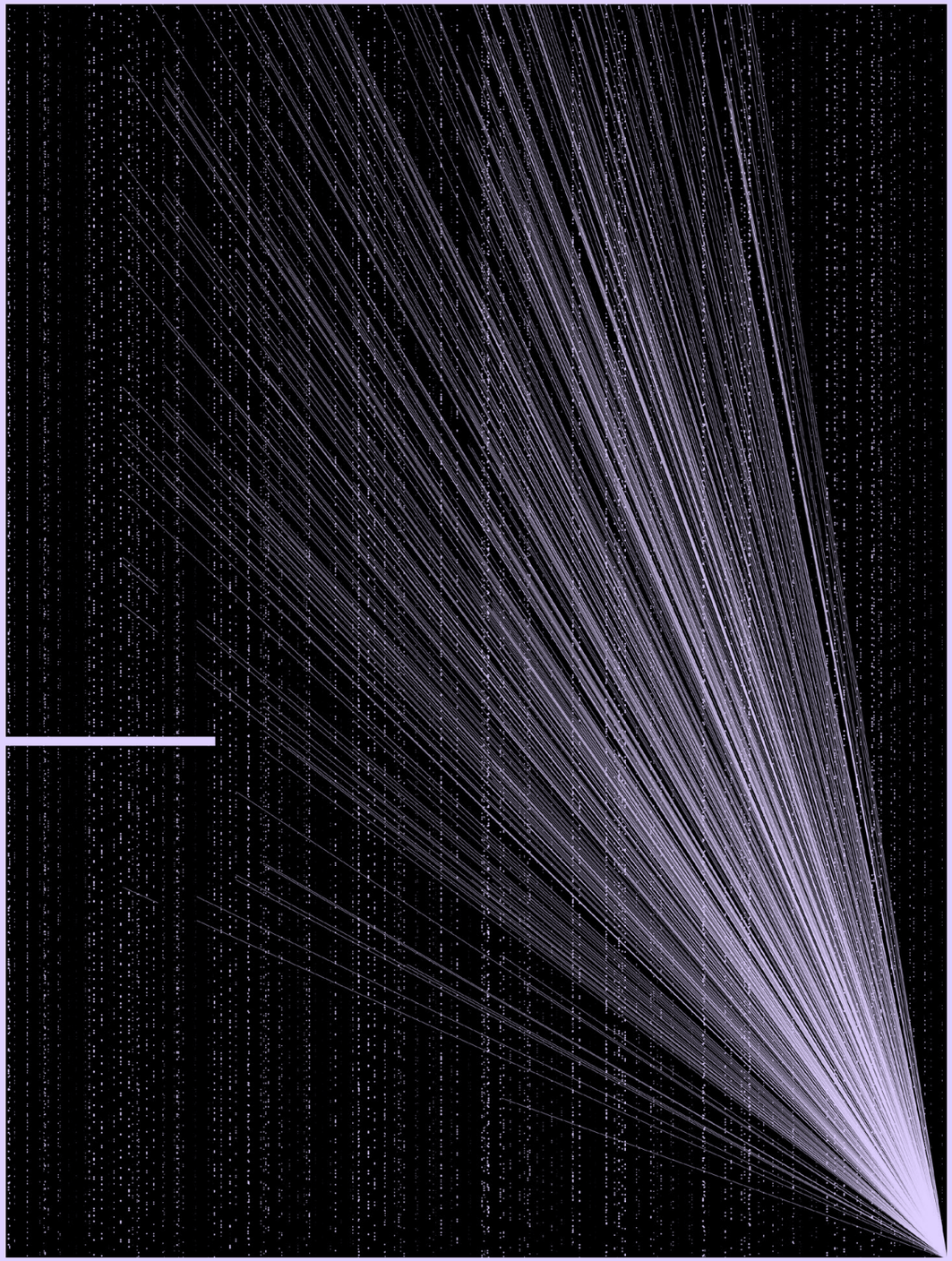
# Langzeit

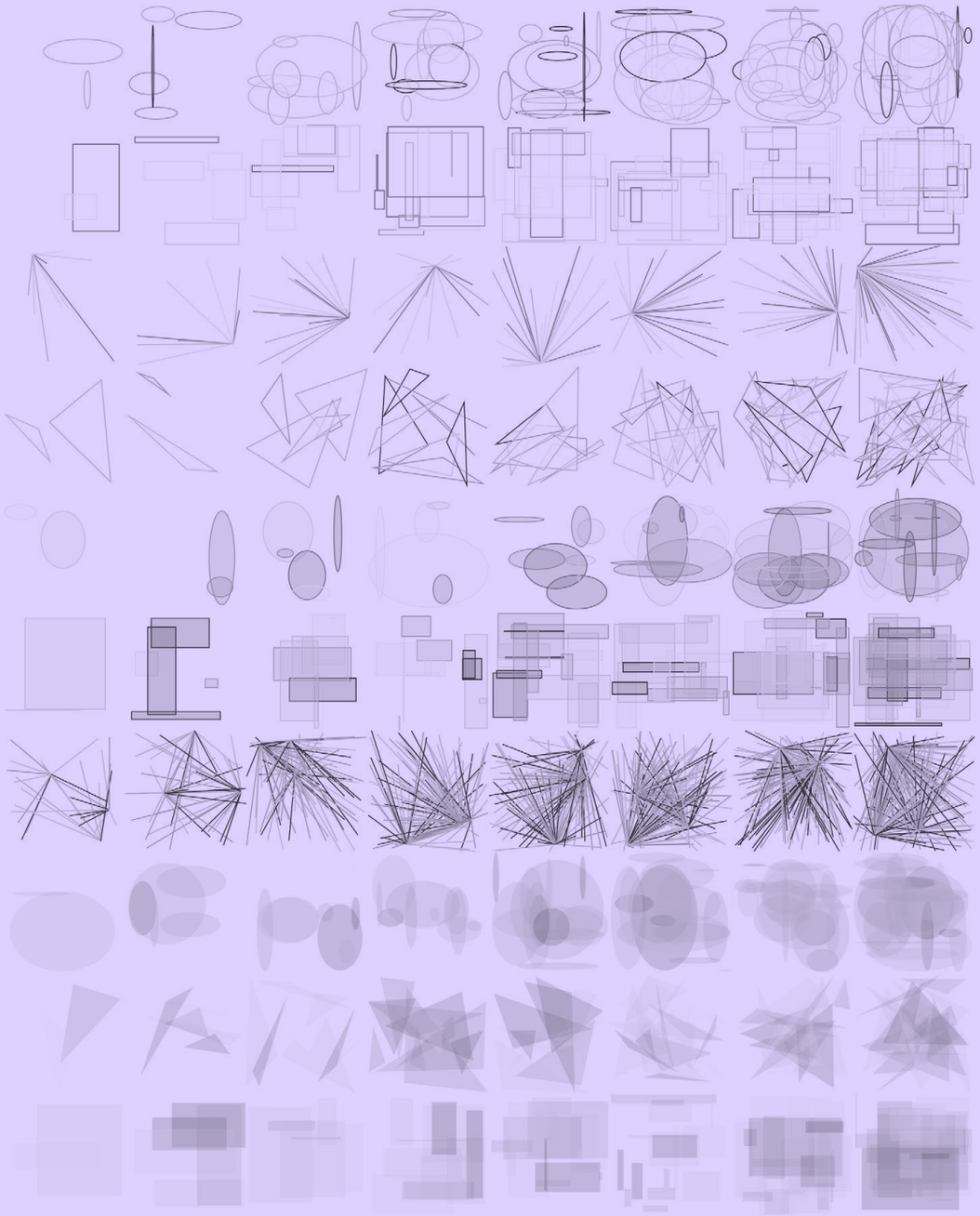
An image
slowly rendered
pixel by pixel
each one
from another point in time

The string of RGB values
not just an image
but a timeline
telling about movement
light
and change

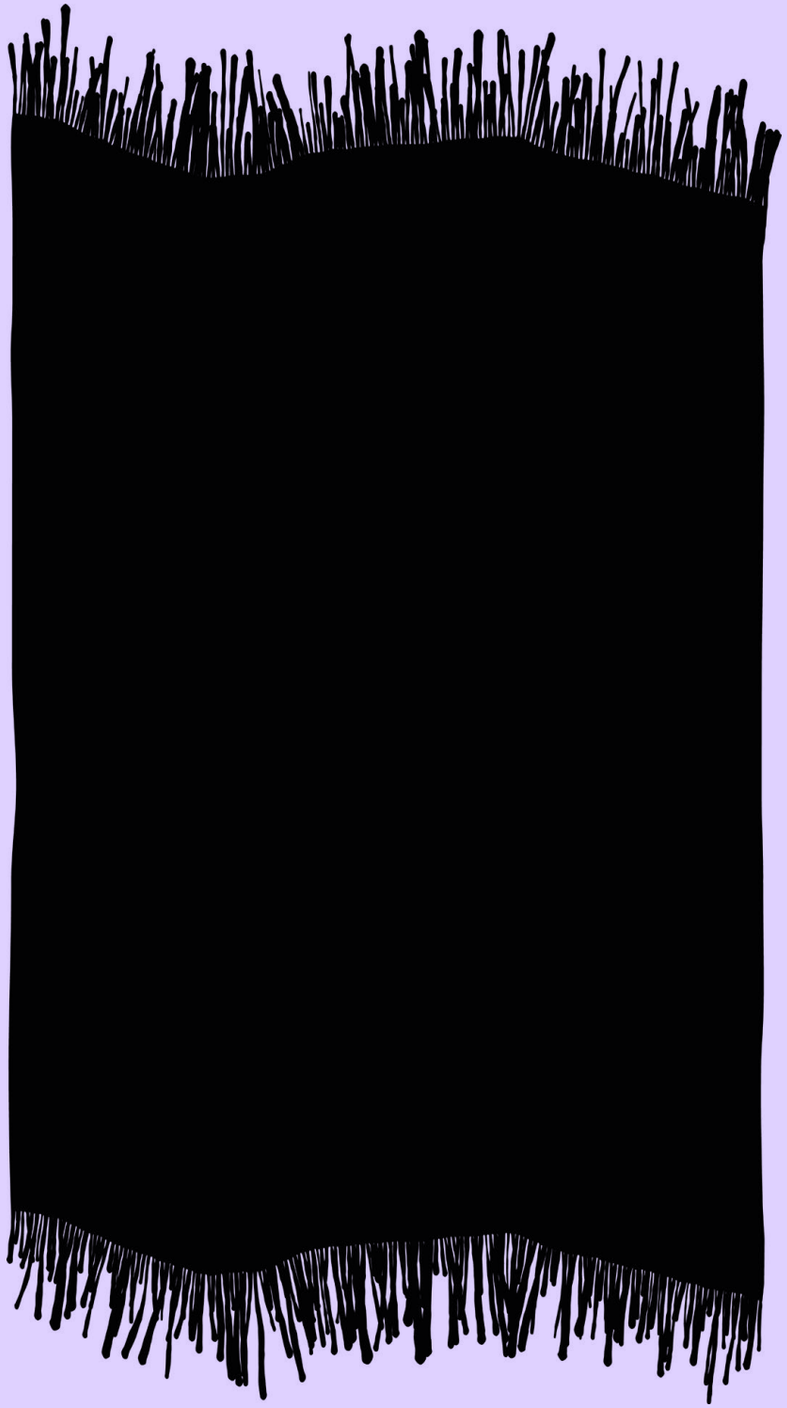
```
void draw() {
    if (cam.available() == true) {
        cam.read();
        cam.loadPixels();
        langzeit.loadPixels();
        langzeit.pixels[x] = cam.pixels[x];
        langzeit.updatePixels();
        x++;
    }
}
```







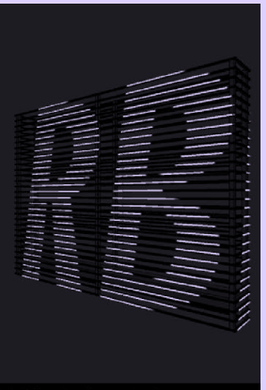




Hej! I am Julian.
I am an avid Processing user and have been using it for the last 12 years — as a tool to funnel ideas, to realize physical media installations, teach coding to new students and explore my own creativity.

ig julian_hespenheide / www.julian-h.de

**Down** Sagenmaschine at GRIMMWELT Kassel, DE with Studio TheGreenEyl (2017).
A visitor is getting analyzed by a Machine Learning app and gets translated into a Brother's Grimm fairy tale.

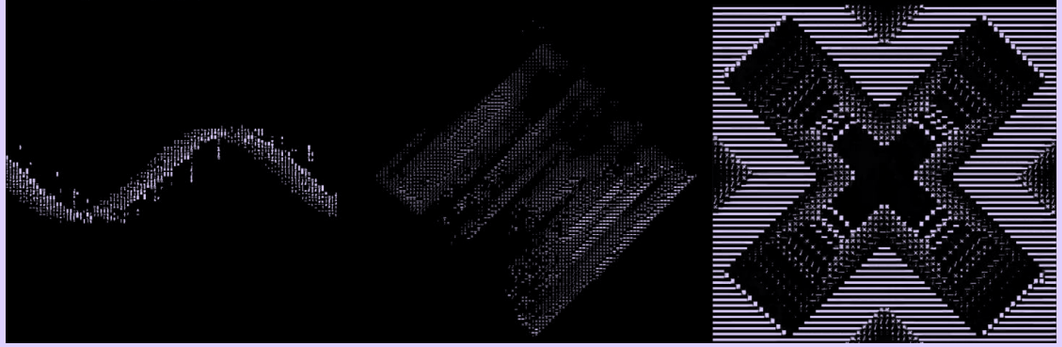


**Left** „Rare moments of light". Running Processing 3.
Permanent exhibition at Neues Bauhaus Museum Weimar, DE with Studio TheGreenEyl (2019)
**Up** Processing Tool with different modes to control the installation.

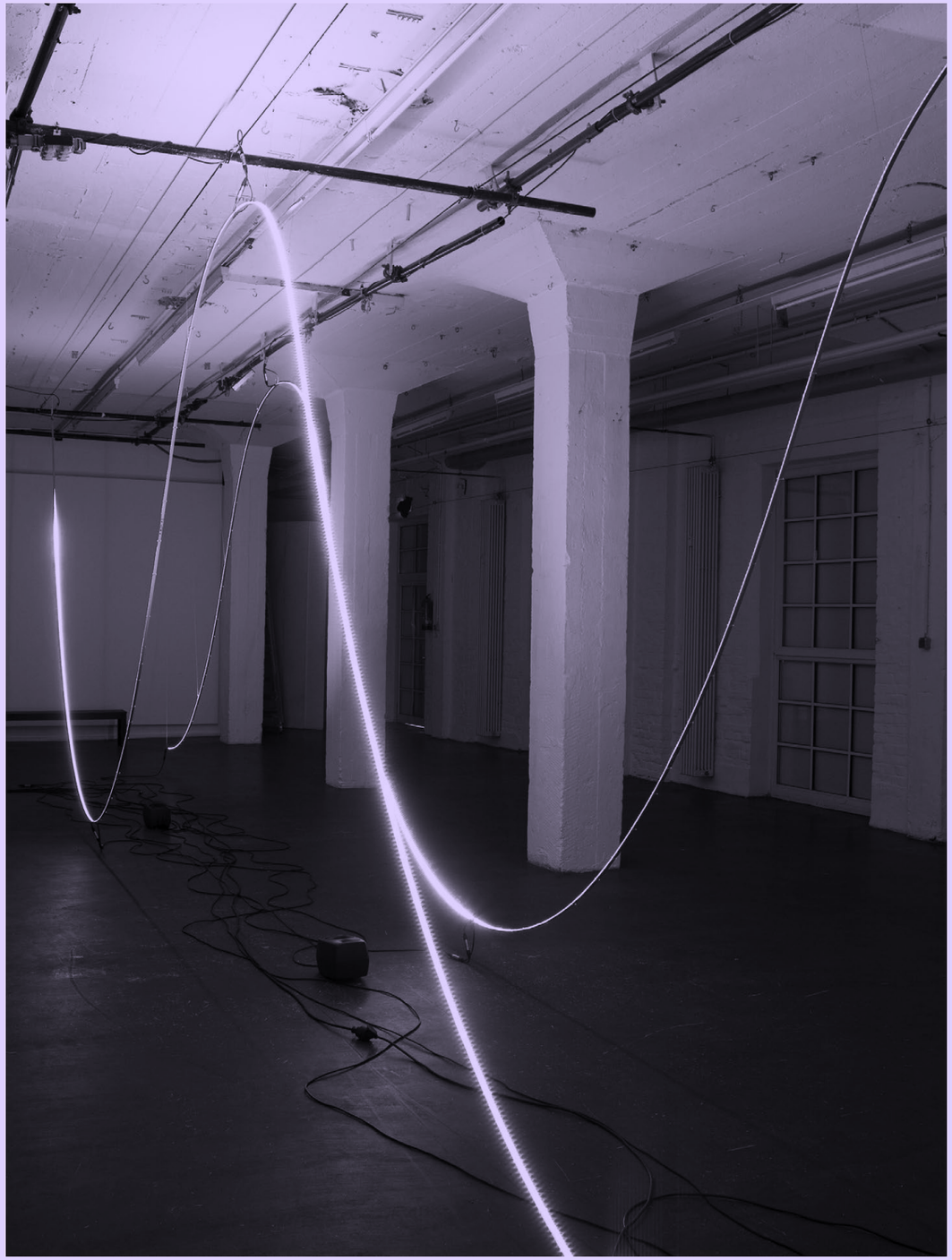
**Up** For Jianping He (graphic designer) his hole body of works of the last 20 years was sorted and displayed with a quad-tree algorithm as an entrance gesture for his solo exhibition.
V&A Museum Shenzhen, China, with Studio TheGreenEyl (2020)

**Left** Staging Absence. New Media Installation. Work for Irena Kukric.
Controlling Curtains, Lights, Sound and Air to stage a play without human actors. (2016)

**Down** Excerpts of an ongoing experimental animation series in 2D with ASCII Characters (2021)



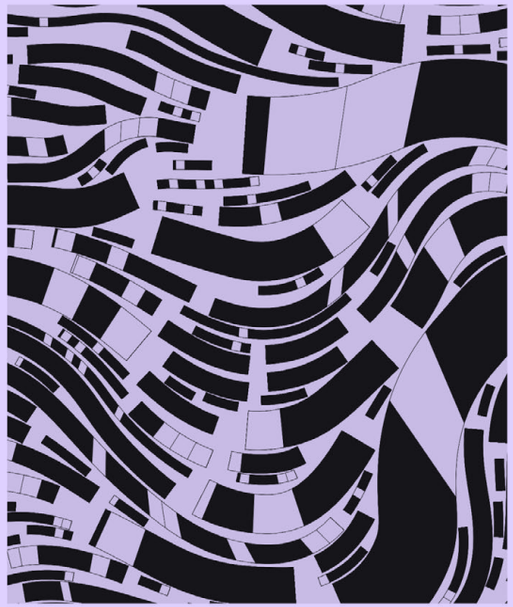


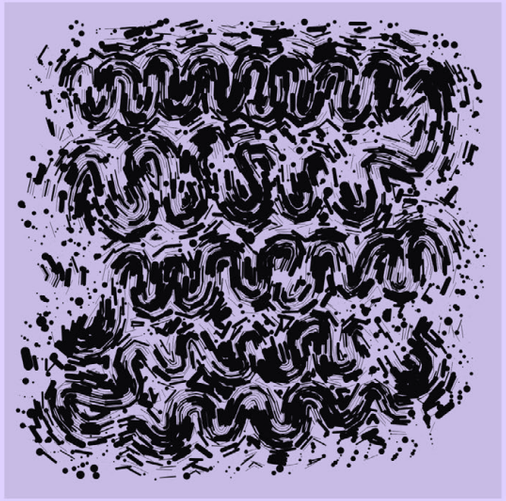


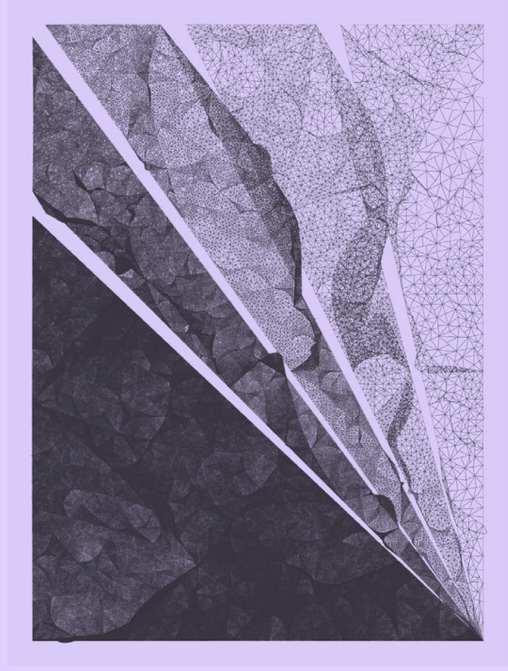
# Imagination by Hevey (@HeveyArt - https://hevey.art)

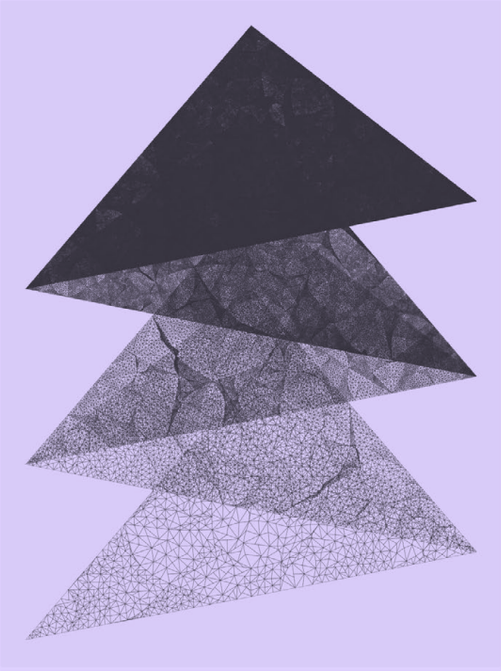




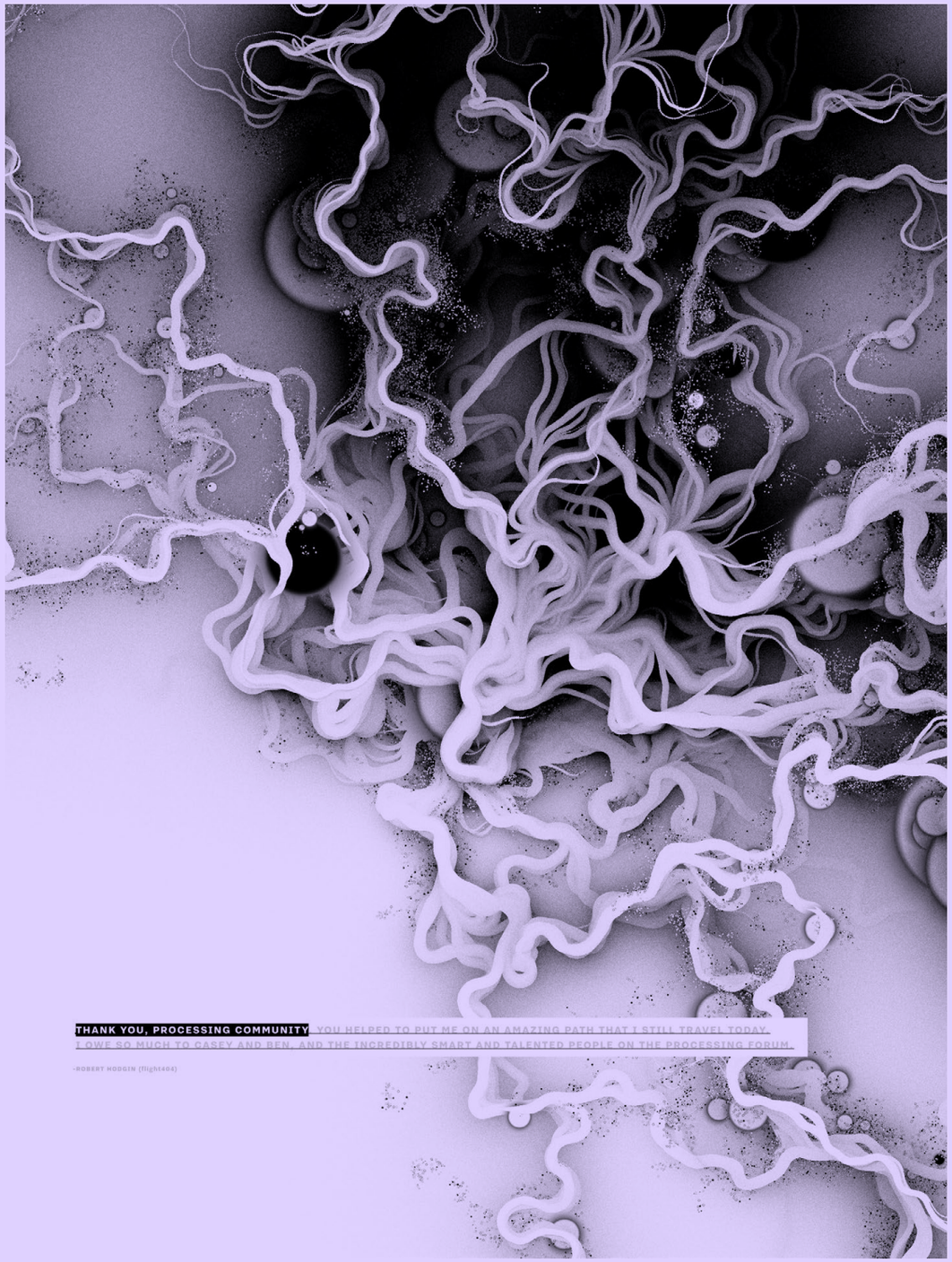




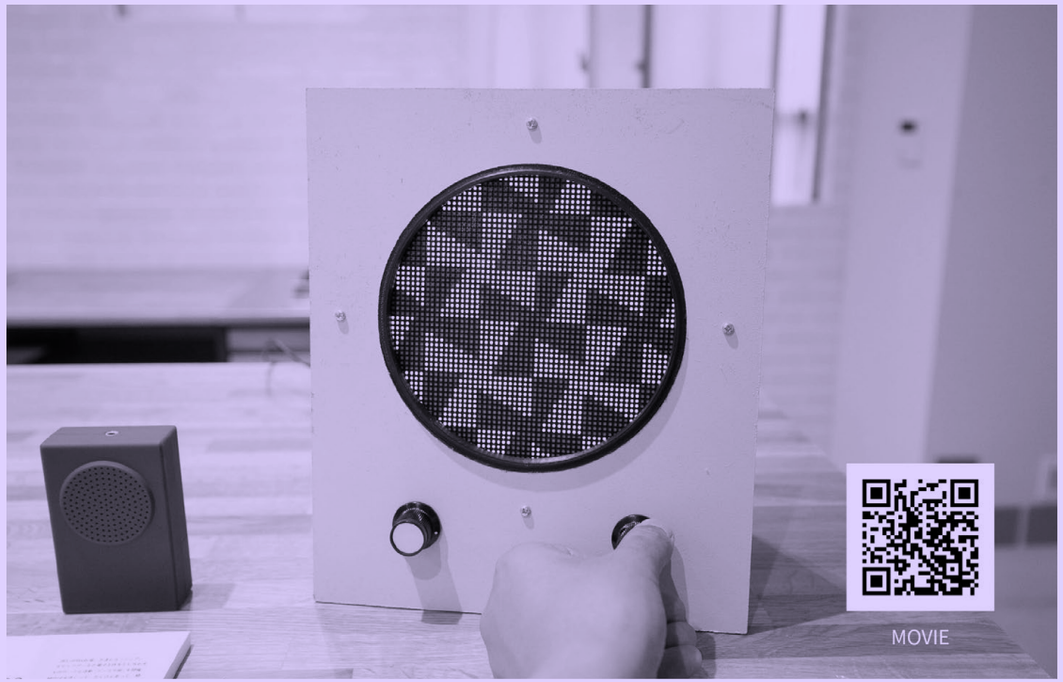




From top left, clockwise: Fidenza #564 (2021), Elektroanima 0.8 (2020), Progress 1A (2018), Progress 1B (2018)





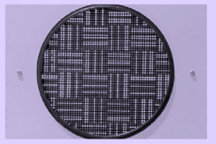

# Buddha Machine Mirage

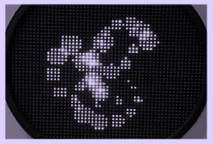
**@hoehoe**

"Buddha Machine Mirage" is a prototype of a machine that projects ambient images. It was developed to be a device that you can watch while listening to ambient music.

The internals consist of a RaspberryPi and a panel of LED modules, which are controlled by Processing. It can be run standalone.

The image displayed by Processing is written in GLSL Shader, and the image pattern changes in a certain amount of time. The parameters can be changed using the knobs on the front of the panel to give changes to the image. The Shader can also be rewritten over the network.



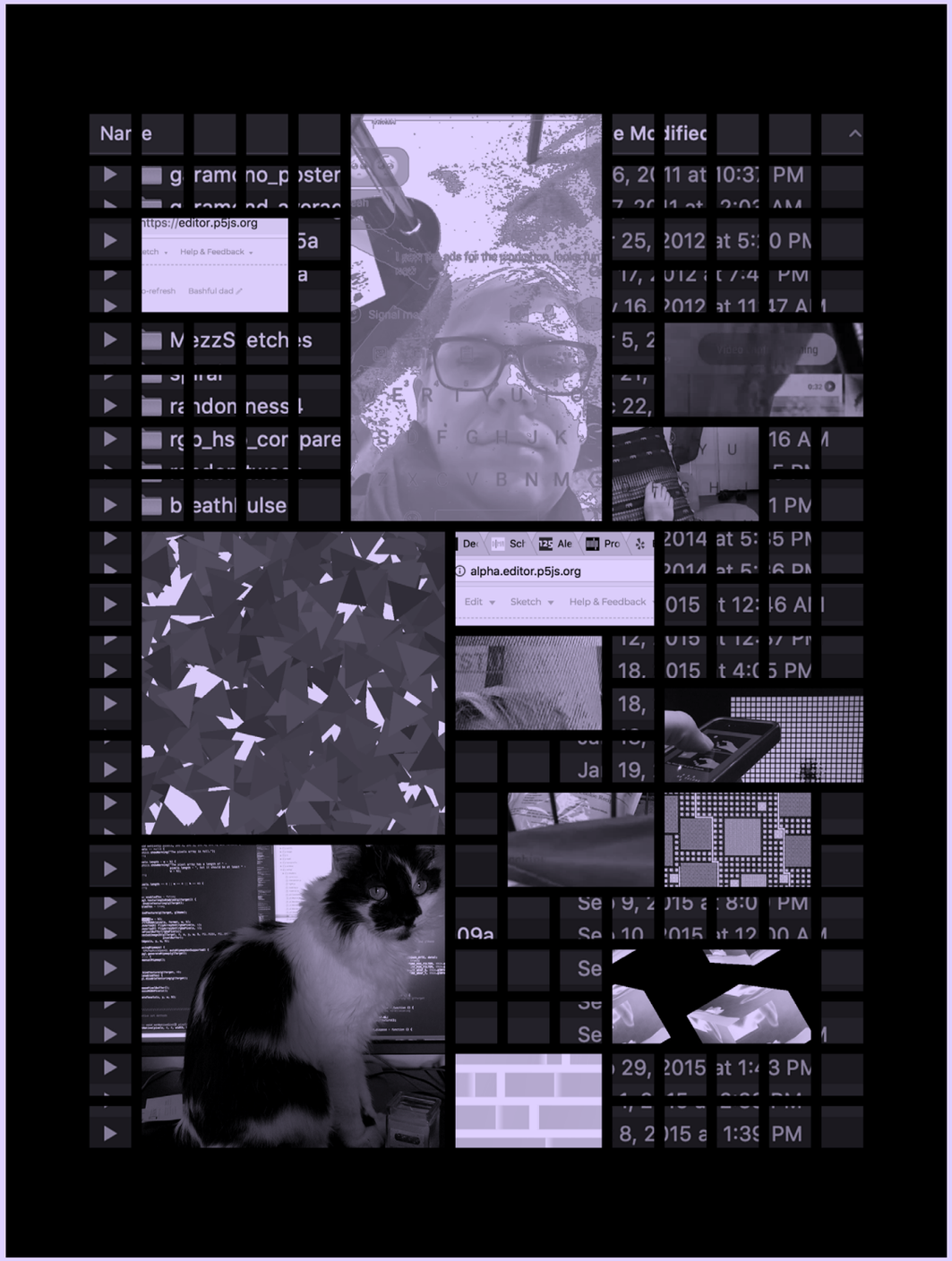


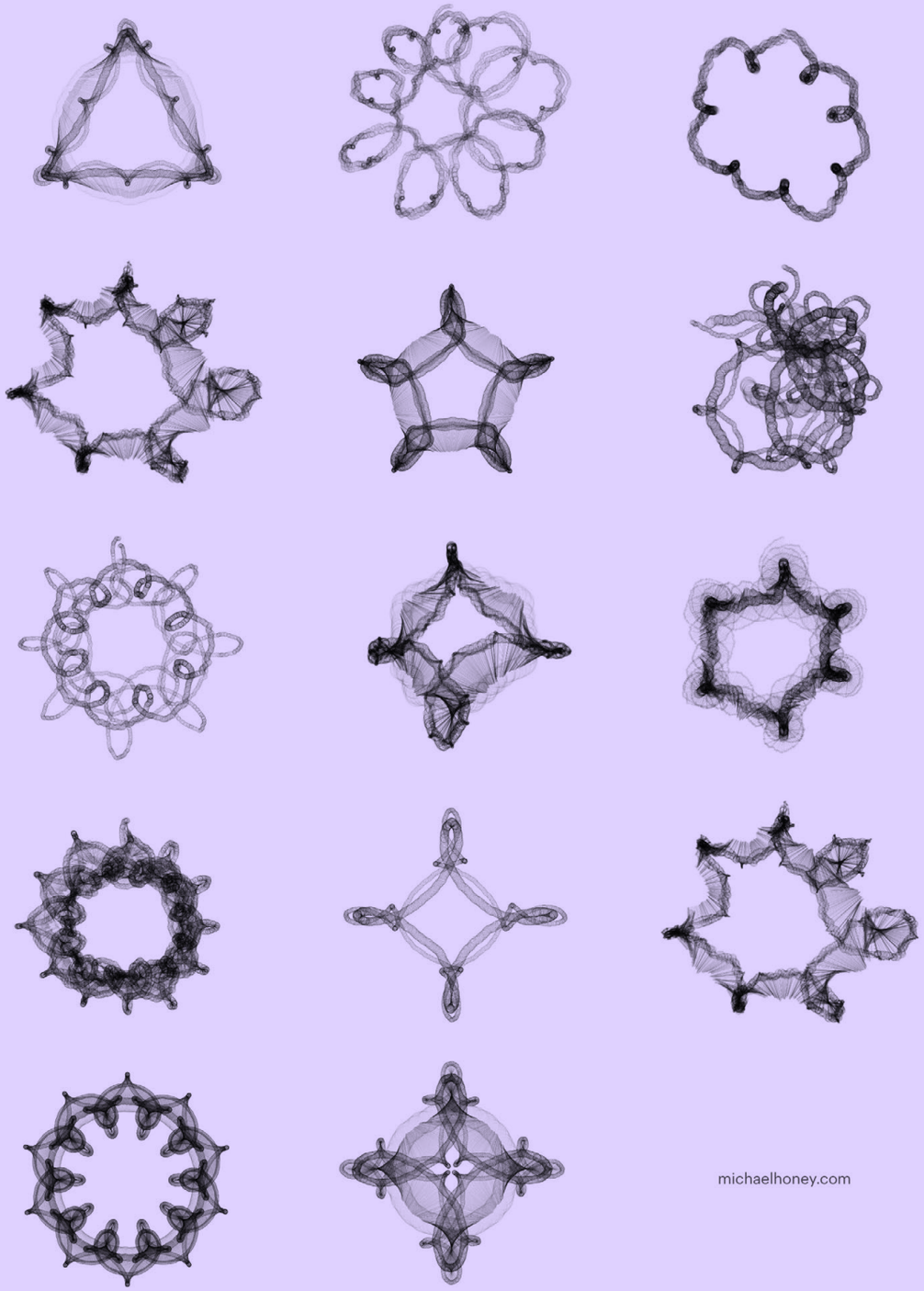
インスタ部
INSTABU

digital art creative community “INSTABU”
https://instabu.net


Sendai
Tokyo
7 days of earthquakes in Japan
09/3/2011 - 16/3/2011

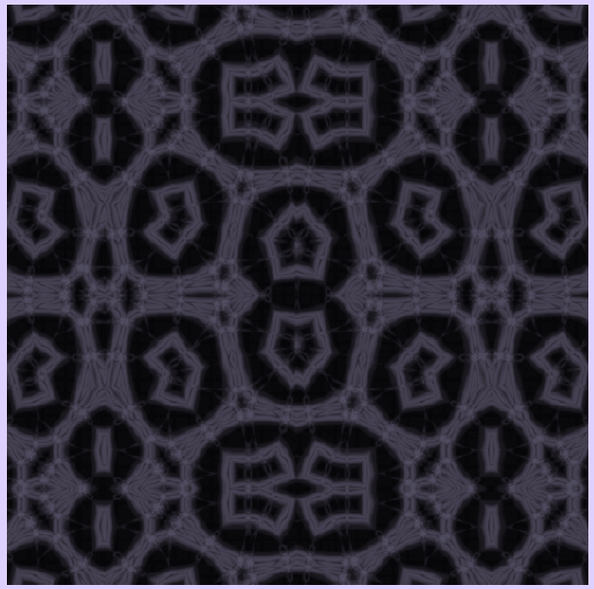



https://editor.p5js.org
alpha.editor.p5js.org
Edit
Sketch
Help & Feedback



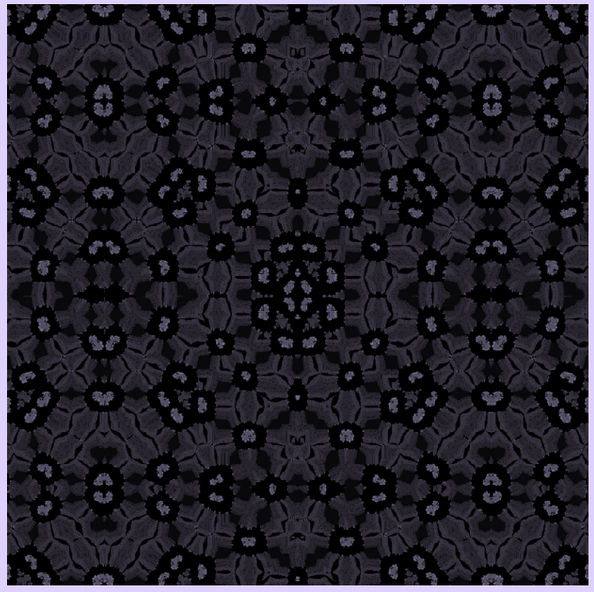



# The cpu and gpu go hand in hand.

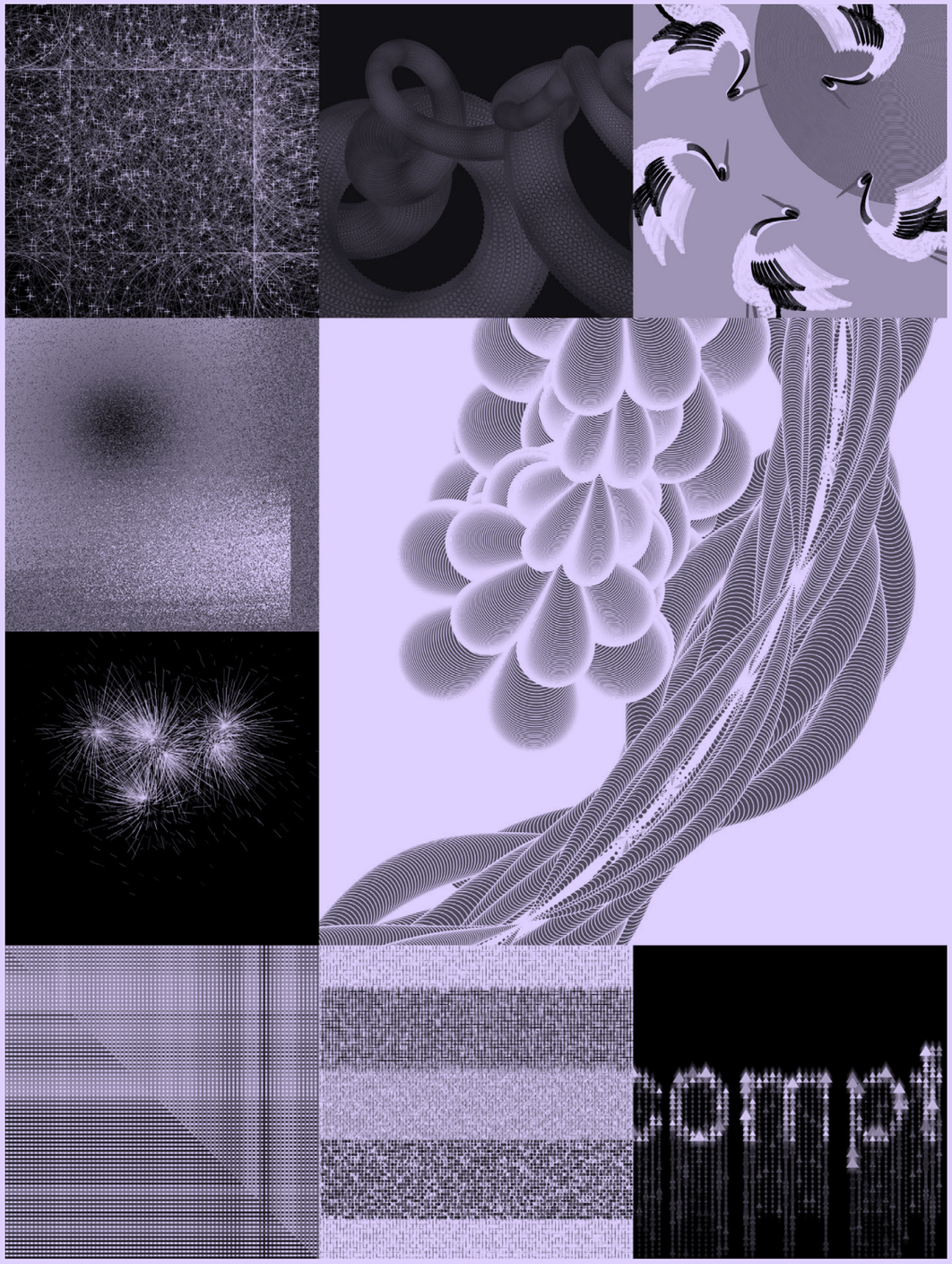


It combines fractals created by Processing with effects created by GLSL.

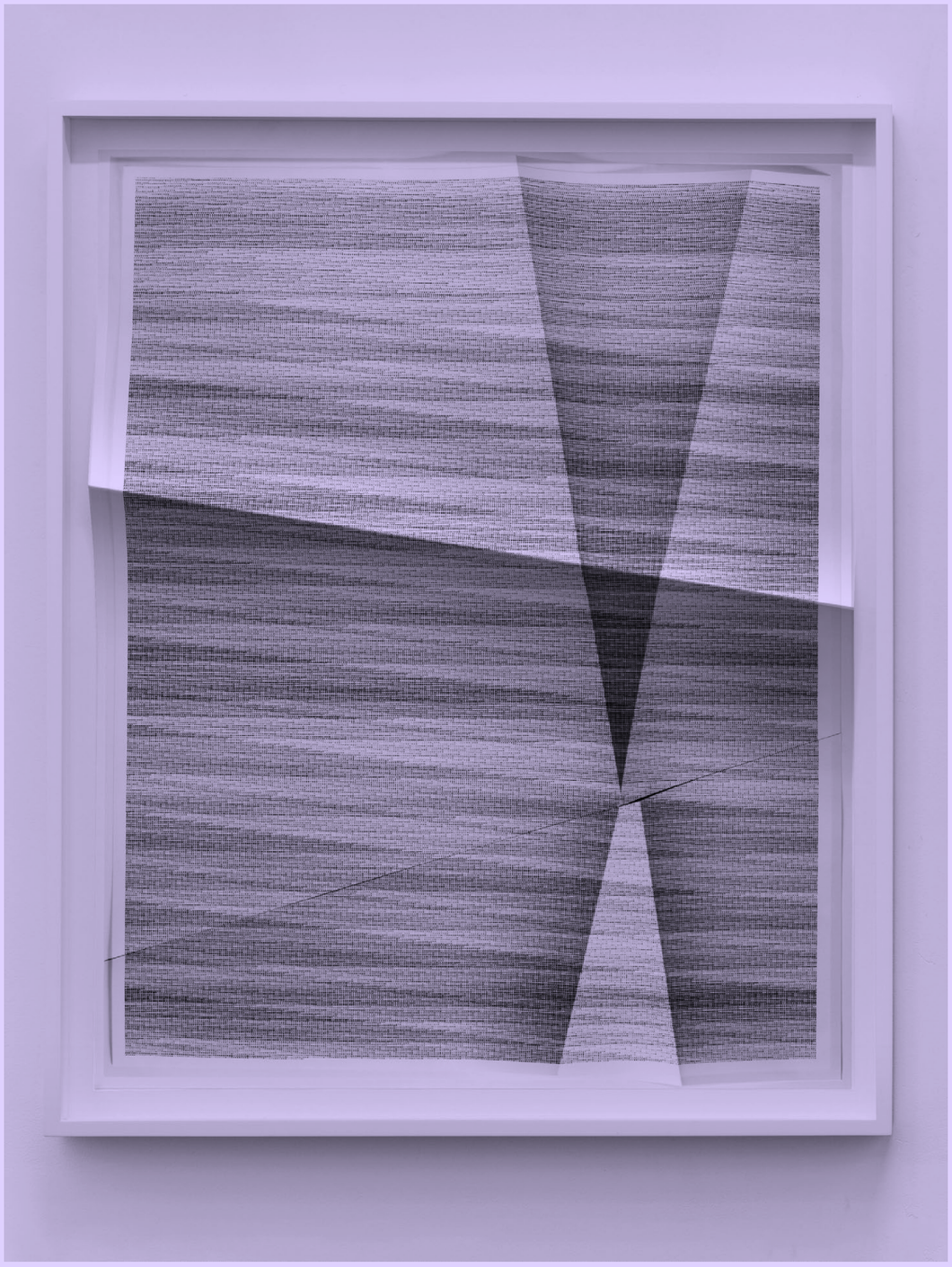
It is a generative process, so it changes variously with time.

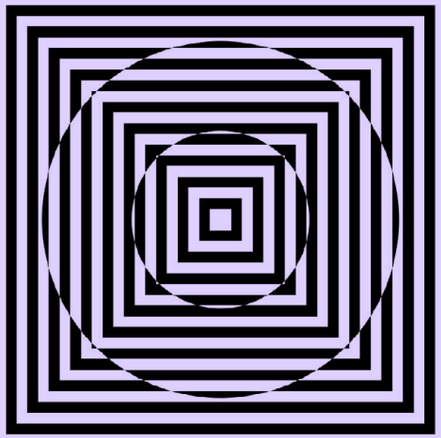
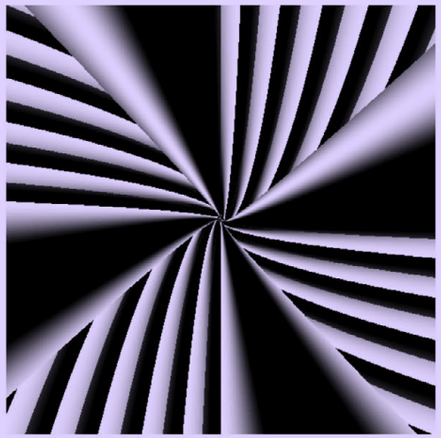
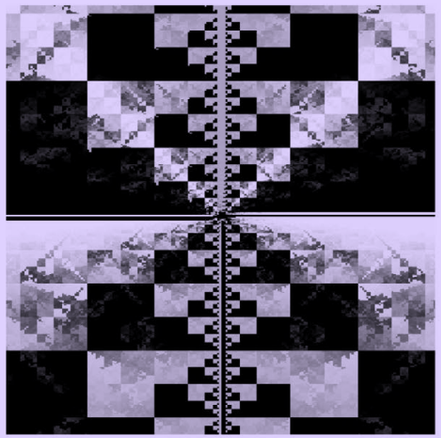
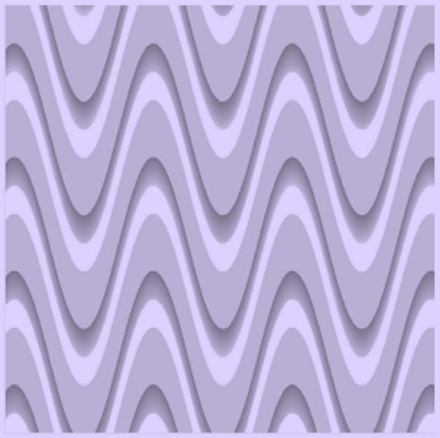
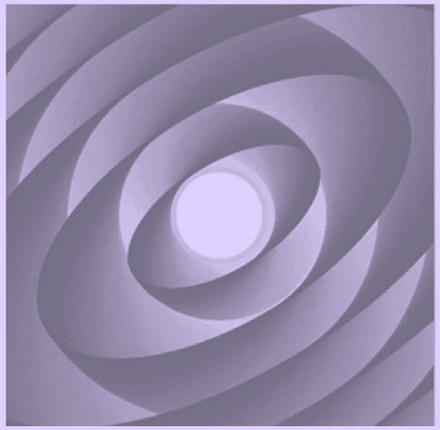
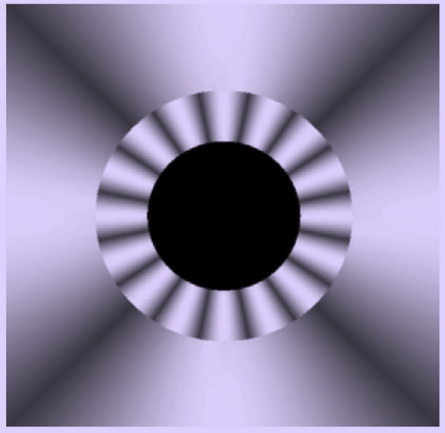


Author
@HosodaMath
Come visit us on Twitter and Github!

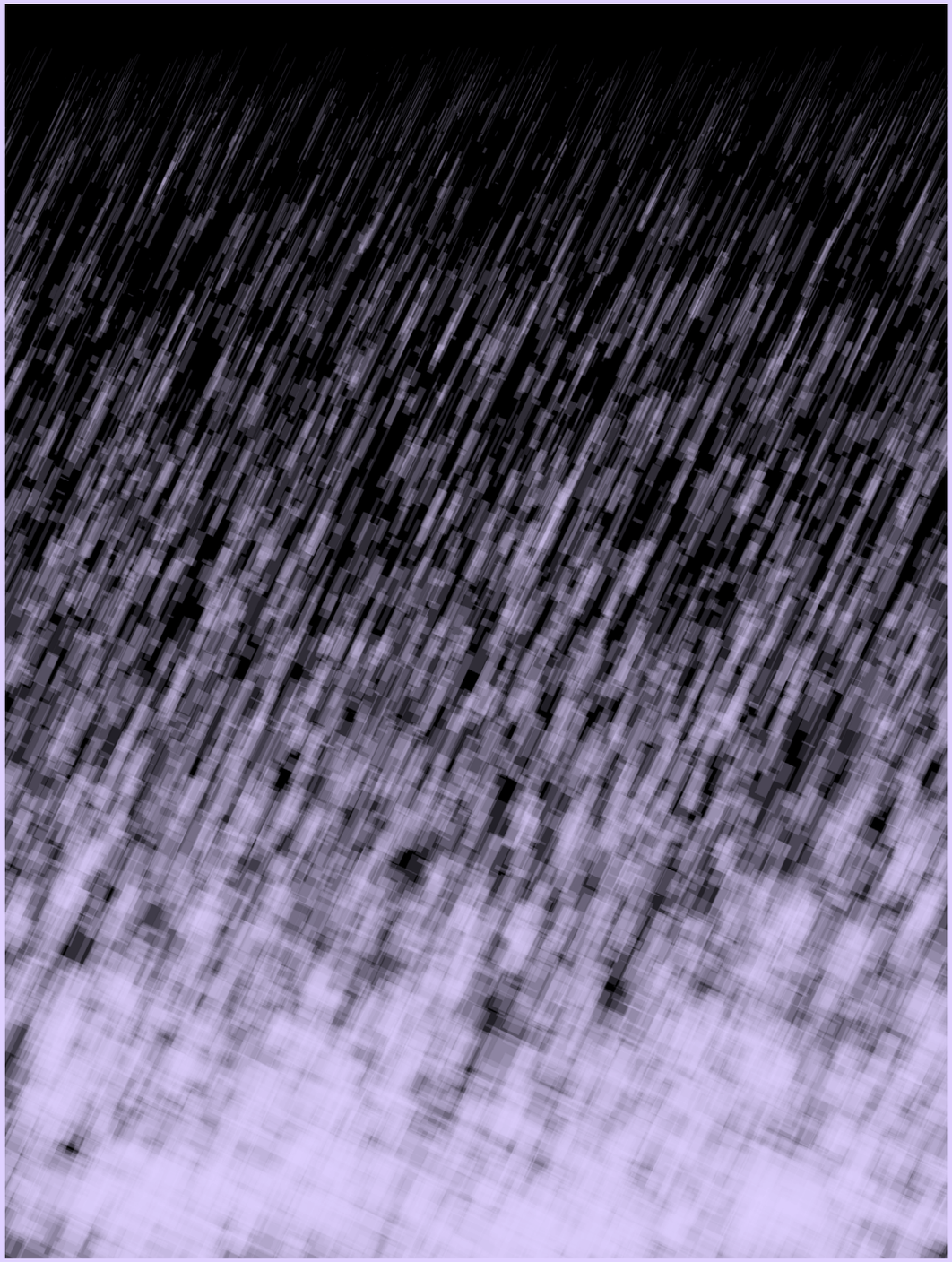




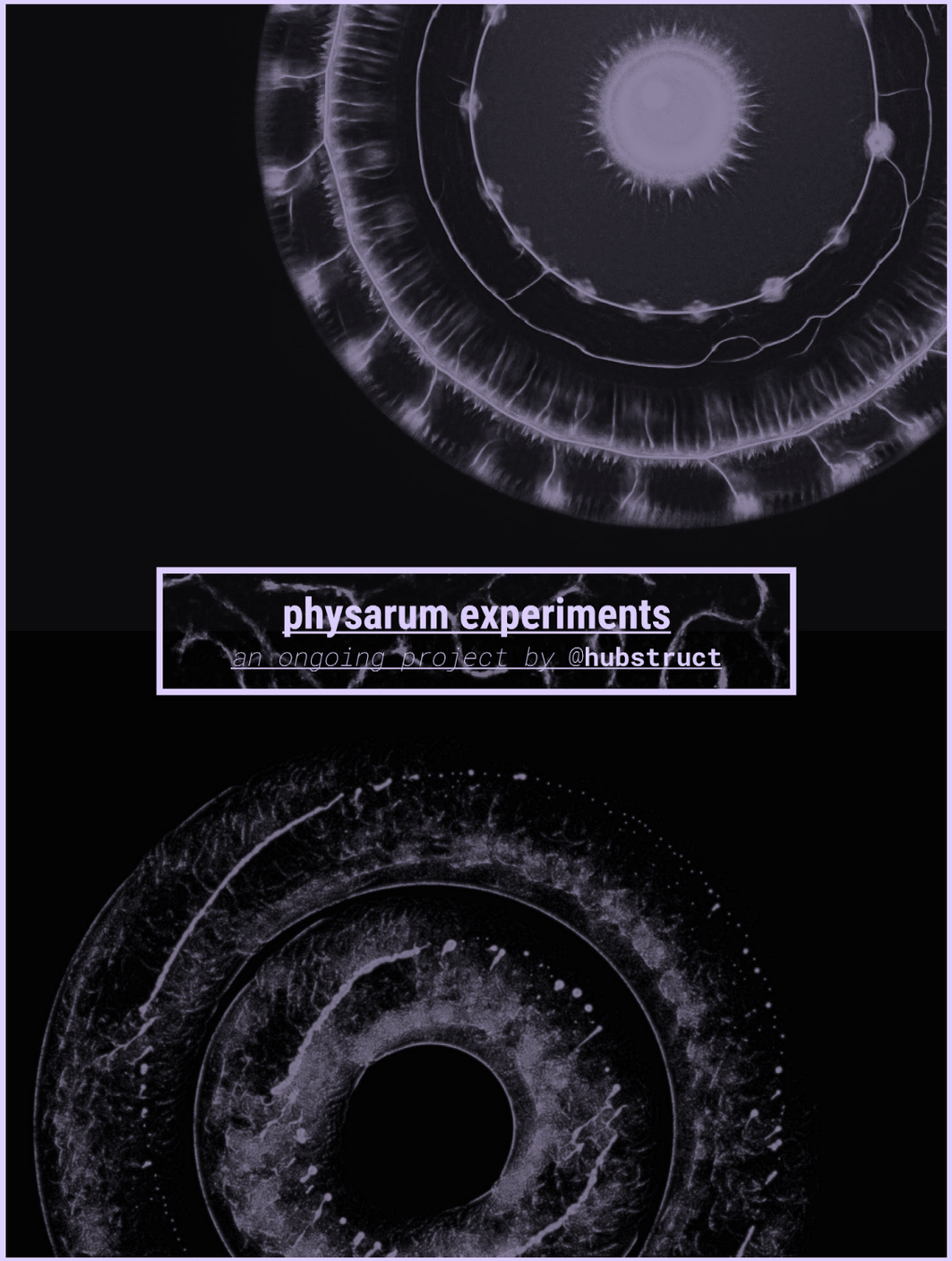


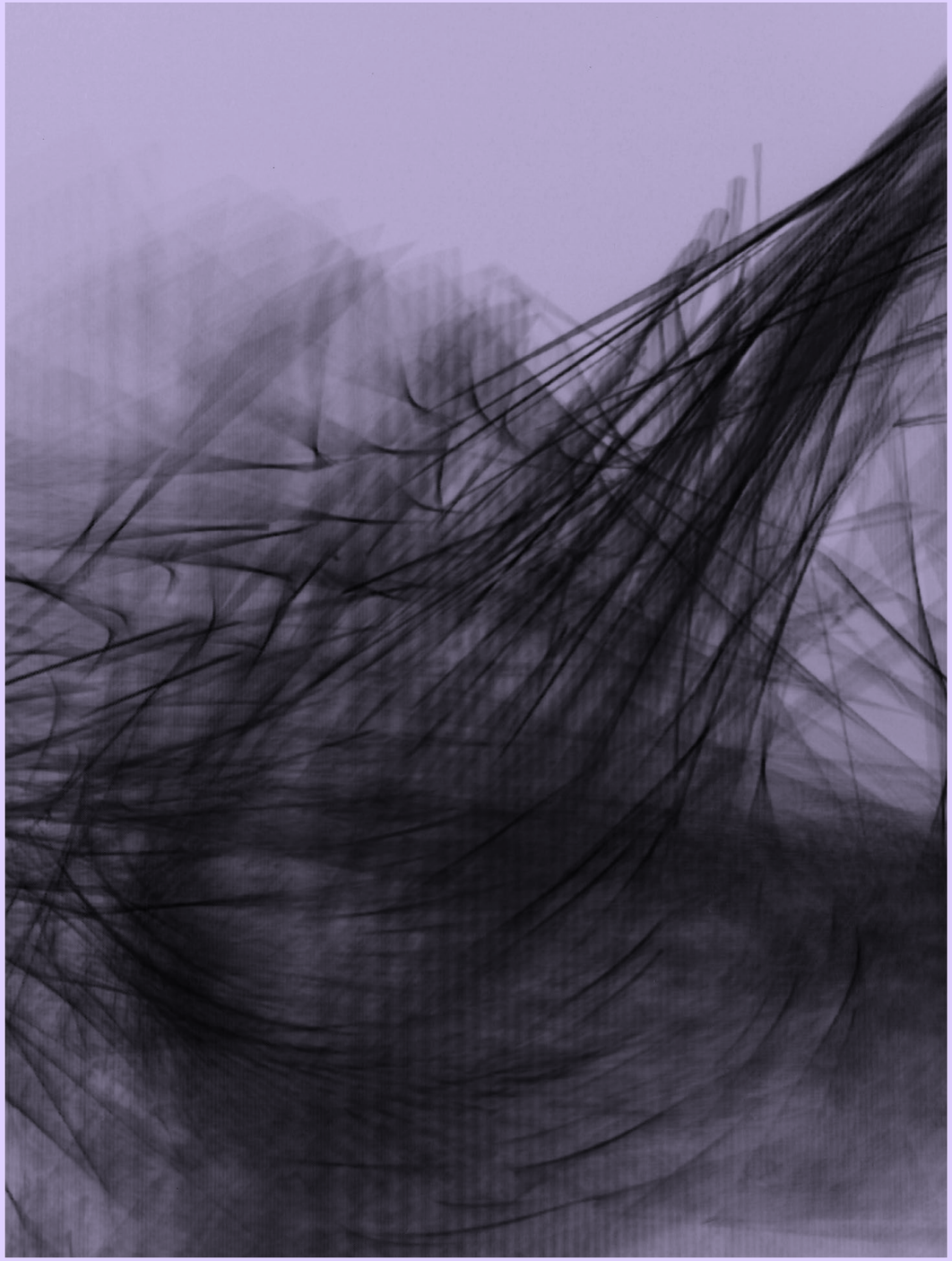














physarum experiments
an ongoing project by @hubstruct

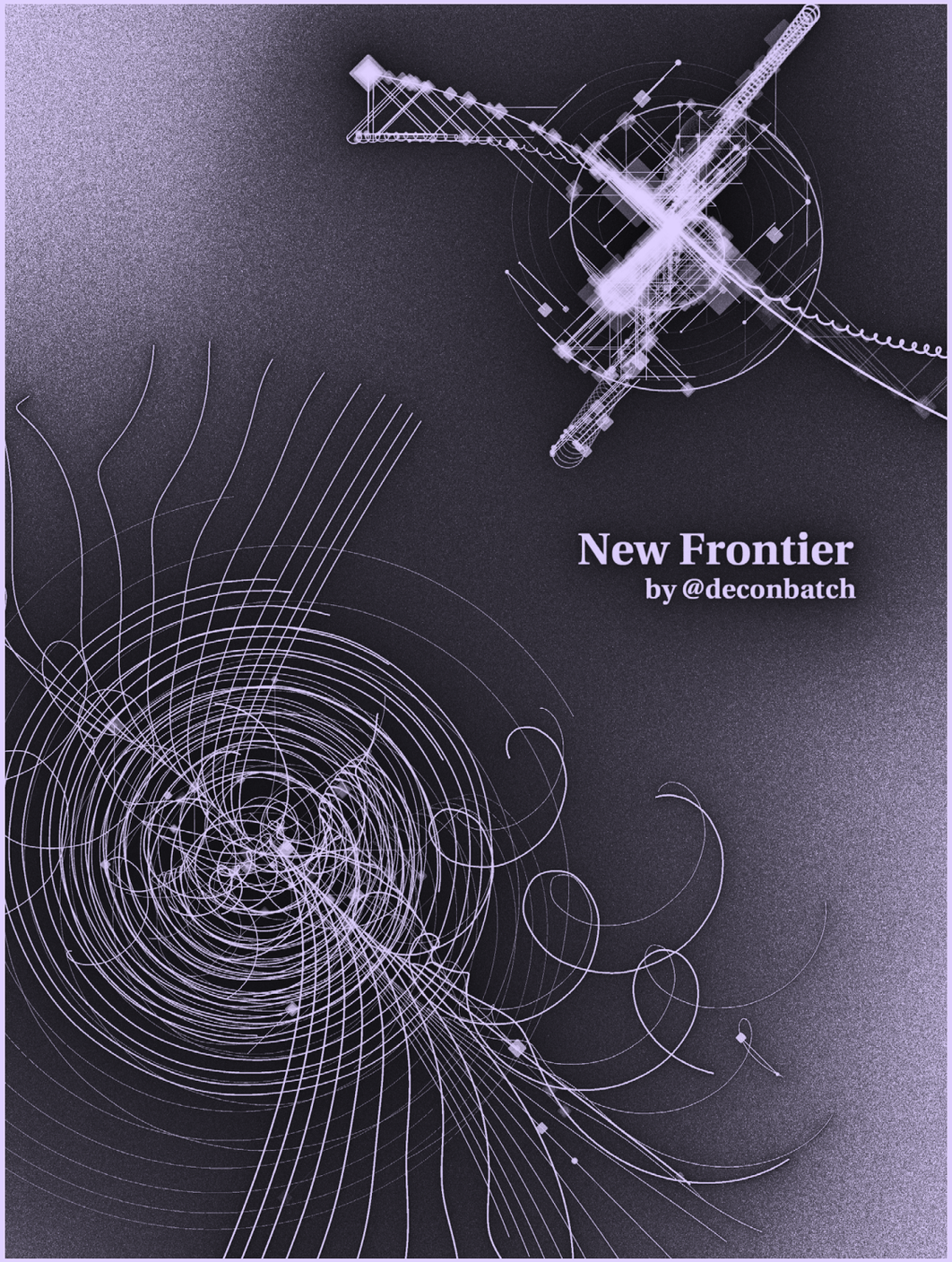




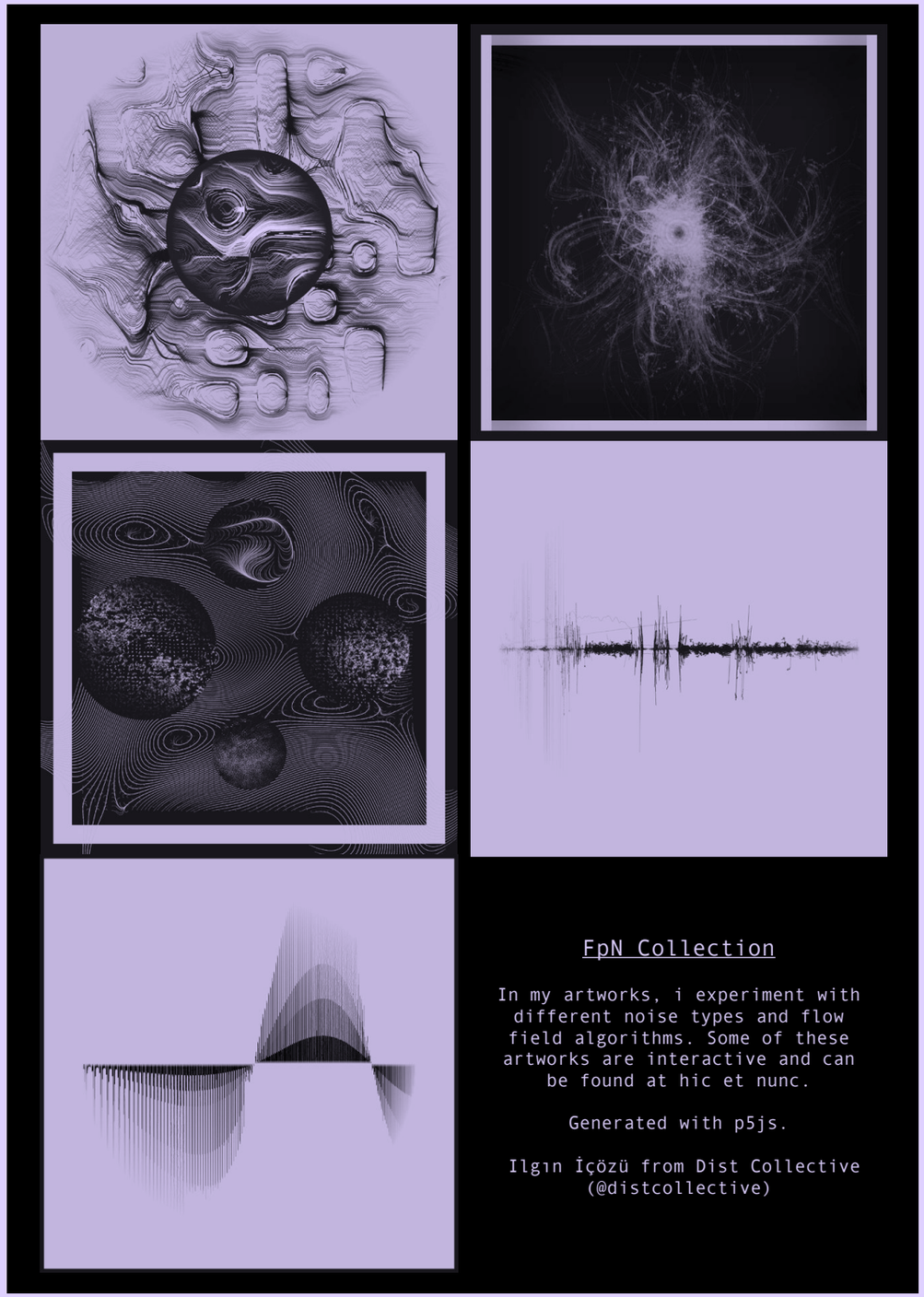
lilcodes
hypereikon lab
sebastián rojas y constanza lobos
agosto, valdivia, chile

//video feedback
//realizado durante PCD 2021 Latinoamerica,
//para Processing × Hic et Nunc fundraiser

```
let seno=(div,min,max,sinmin=-1,sinmax=1)=>map(sin(frameCount/div),sinmin,sinmax,min,max);letlastImg;functionsetup(){createCanvas(windowWidth,w
indowHeight);background(255,225,250);zoom=1};functiondraw(){if(lastImg){image(lastImg,-1*zoom,-1*zoom,width+2*zoom,height+2*zoom)};lastImg=get(
);max_i=15;for(let i=1;i<max_i;i++){push();blendMode(BLEND);rotate(i/max_i);translate(seno(-(2*width)/(i*20),0,width),0);image(lastImg,;i*width
/20,i*width/30,;i*width/10,height/30,;i*width/20,i*width/30,;i*width/100,height/30);pop()};push();blendMode(BLEND);translate(width/2,height/2);
scale(seno(6.666,1,seno(18.333,1,1.5)));rotate(frameCount/70);rotate(frameCount/100);translate(-width/2,-height/2);stroke(seno(30,125,350),senc
(130,75,200),seno(1/3.333,100,110),seno(100,220,255));strokeWeight(width/50);fill(seno(40,0,255),seno(45,0,155),seno(50,0,55),3);translate(senc
(7,-width/30,width/30),seno(10,-height/30,height/30));ellipse(width/2,width/2,seno(50,-width/3,width/3)+seno(35,-width/2,width/3),100);pop();pu
sh();blendMode(BLEND);noStroke();fill(255,seno(33.3*(frameCount/10)%int(seno(14,3,12)+1),25,120),seno(20,0,100),(frameCount%255-frameCount%175)
*(frameCount%5+1));translate(width/2,height/2);rotate(frameCount/-17);translate(-width/2,-height/2);rect(seno(12,width/6,5*width/6),seno(18,hei
ght/6,5*height/6),width/2,10,30);pop();push();noStroke();blendMode(BLEND);fill(seno(53.33,125,250),(frameCount/10)%255,(frameCount/1.666)%255);
translate(width/2,height/2);scale(seno(4,.125,.5));rotate(frameCount/16);translate(width/-2,height/-2);rotate(frameCount/10);rotate(frameCount/
-29);rect(seno(30,0,width/2),seno(47,0,height,-.51),width/seno(3,4,40),height/map(tan(frameCount/2.75),-1,1,4,40));pop();lastImg=get()};functic
nwindowResized(){resizeCanvas(windowWidth,windowHeight)};
```





FpN Collection

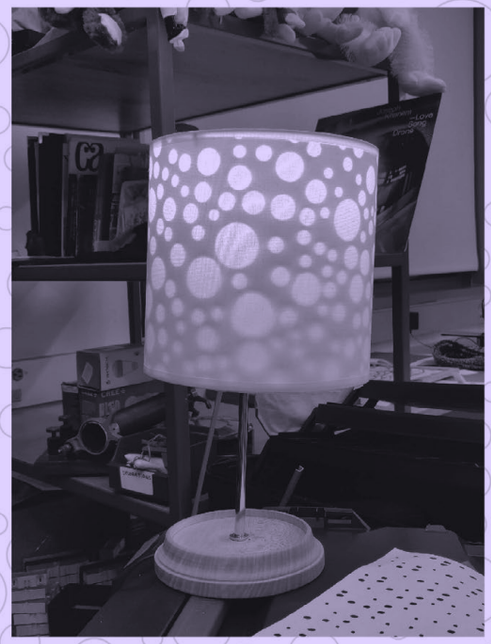
In my artworks, i experiment with different noise types and flow field algorithms. Some of these artworks are interactive and can be found at hic et nunc.

Generated with p5js.

Ilgın İçözü from Dist Collective (@distcollective)



I've made a lot of sketches with Processing and p5.js over the years, but some of my favorites are physical things that are generated by sketches.



# Anti-dot lamp

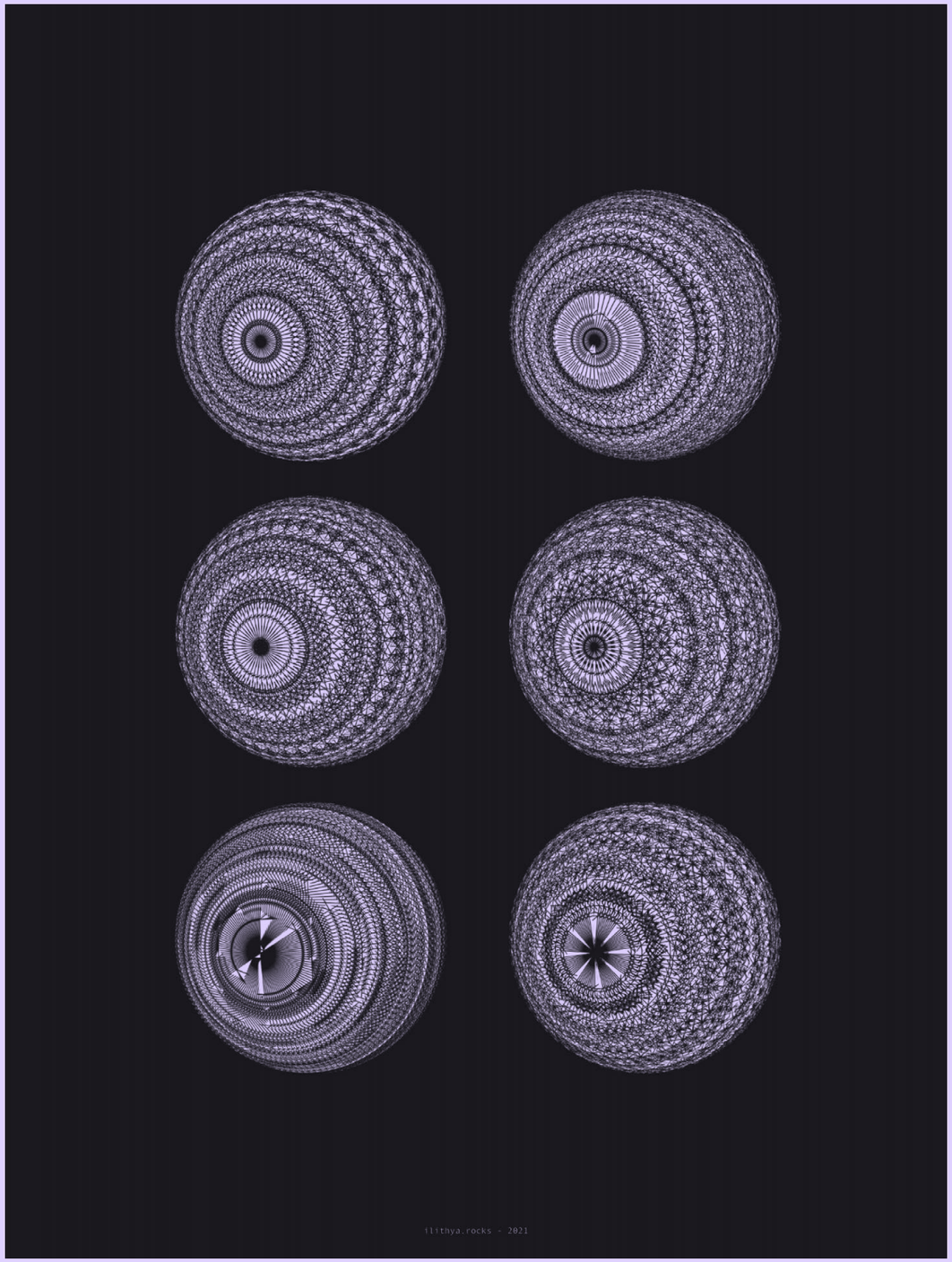
This lamp is one of my current favorites. It's a bit wonky and not that well-made, because I put it together in about an hour. But it makes me happy because it's a physical thing that began as a piece of software, and was made spontaneously and quickly. Most folks look at it and have no idea that it was ever software, and that's how it should be, to me. The p5.js sketch from which it came was just a means to an end.

The lamp is cobbled together from parts from lamp stores. To create the pattern, I modified a circle-packing algorithm from one of Dan Shiffman's tutorials. I added Zeno Zeng's SVG library. To get more context for SVGs, I looked at Processing's SVG and DXF export libraries and Rune Madsen's rune.js, all in the Processing family.

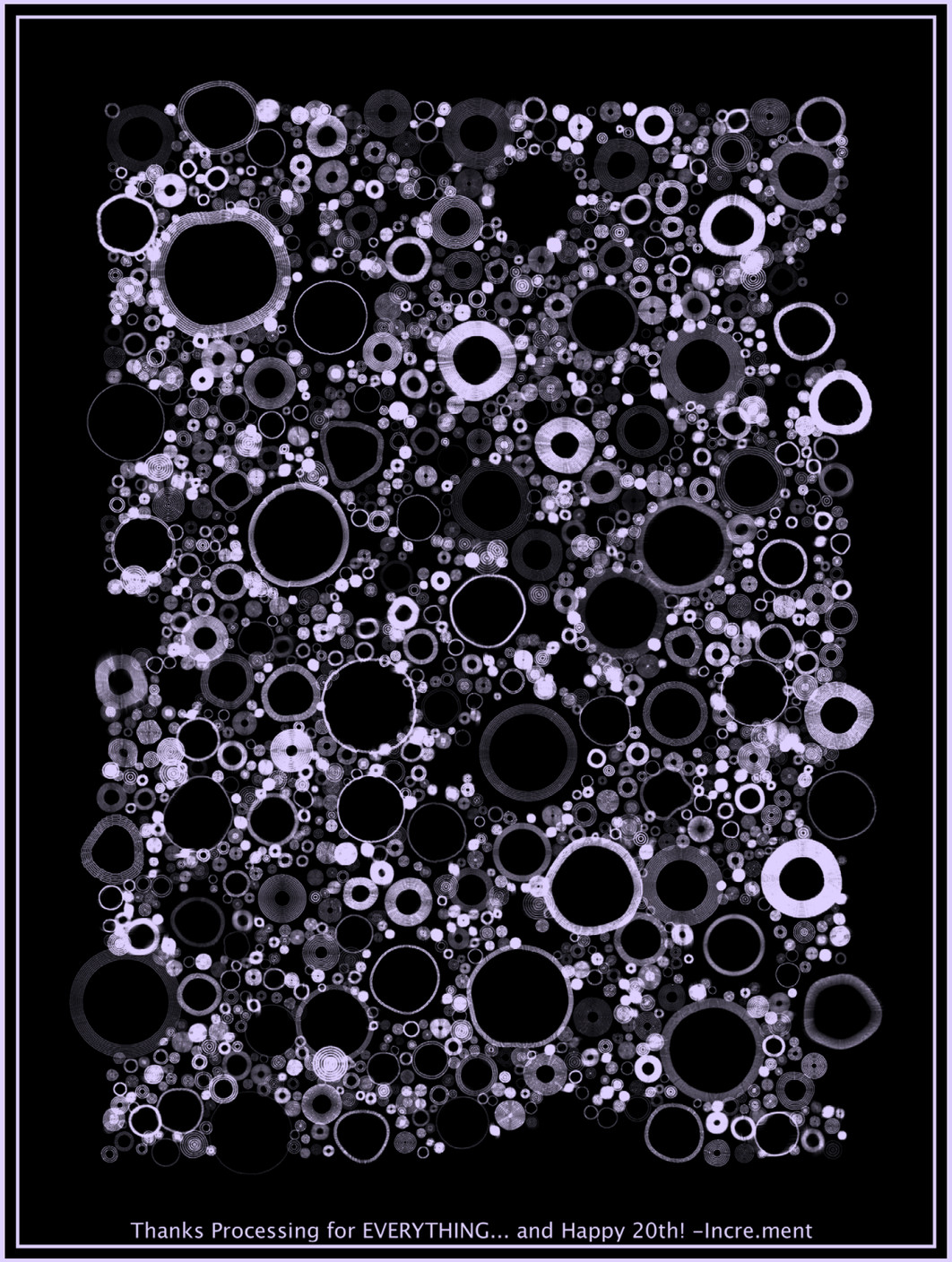
The output was a sketch that generated an SVG file that I could take to the laser cutter with a sheet of construction paper, and the job was done. When my colleague Danny Rozin saw it, he dubbed it "the anti-dot".

I made this for a lighting class, to show students that even if they only had experience from an intro programming class, they could still use those skills to make and customize physical things. To me, that has always been the value of Processing and the tools it's inspired -- not just tools to make images, but to help people to realize how they can affect their world.

*-Tom Igoe*

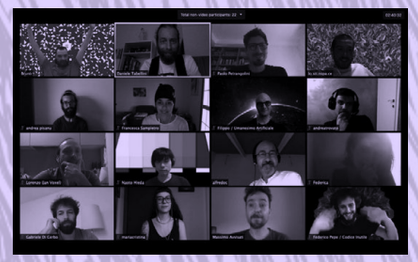



PSYCHEDELIC EYE BY ILITHYA @ILITHYA_ROCKS CON 245 560




INCRE.MENT CON 246 561



**Processing Community Hangout 2020**

**Processing Community Day 2019 Milan**

**Processing Community Day 2019 Rome**

**Processing Community Day 2019 Bari**

*www.codiceinutile.org*
*@codiceinutile*

# SIGNOS

This is an experimental project developed with p5js and ml5js. SIGNOS is a web-browser based video recognition platform that recognizes libra (brazilian sign language) signs and translates them into written words.

The main goal was to make videocall conversation less unilateral. People who can speak have several subtitling tools to have their way of communicating translated for those who can't hear, but the opposite is not that common.


This MVP version of the sign-translator contains 3 signs: "Hello", "I understand" and "Query", respectively "Oi", "Entendi" e "Dúvida". The subtitled video can be sent to platforms like OBS Camera and then streamed to any videocall app such as Zoom and Google Meet.

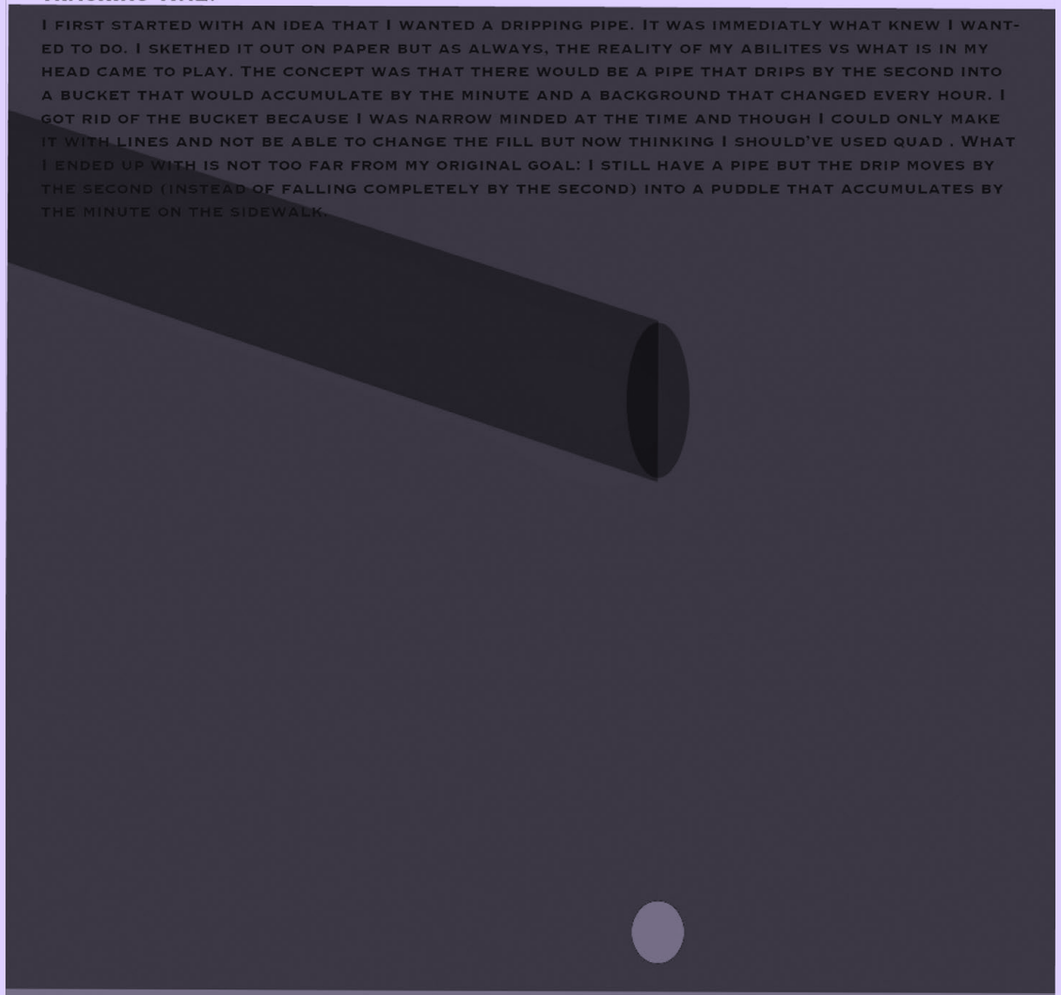
If you'd like to know more about this project please contact me at @isabelazamith.



## EXPERIMENTAL CLOCK | MARCINIA JACQUET

**PROMPT:DESIGN AND CODE AN EXPERIMENTAL CLOCK THAT EXPRESSES A NOVEL WAY OF TRACKING TIME.**

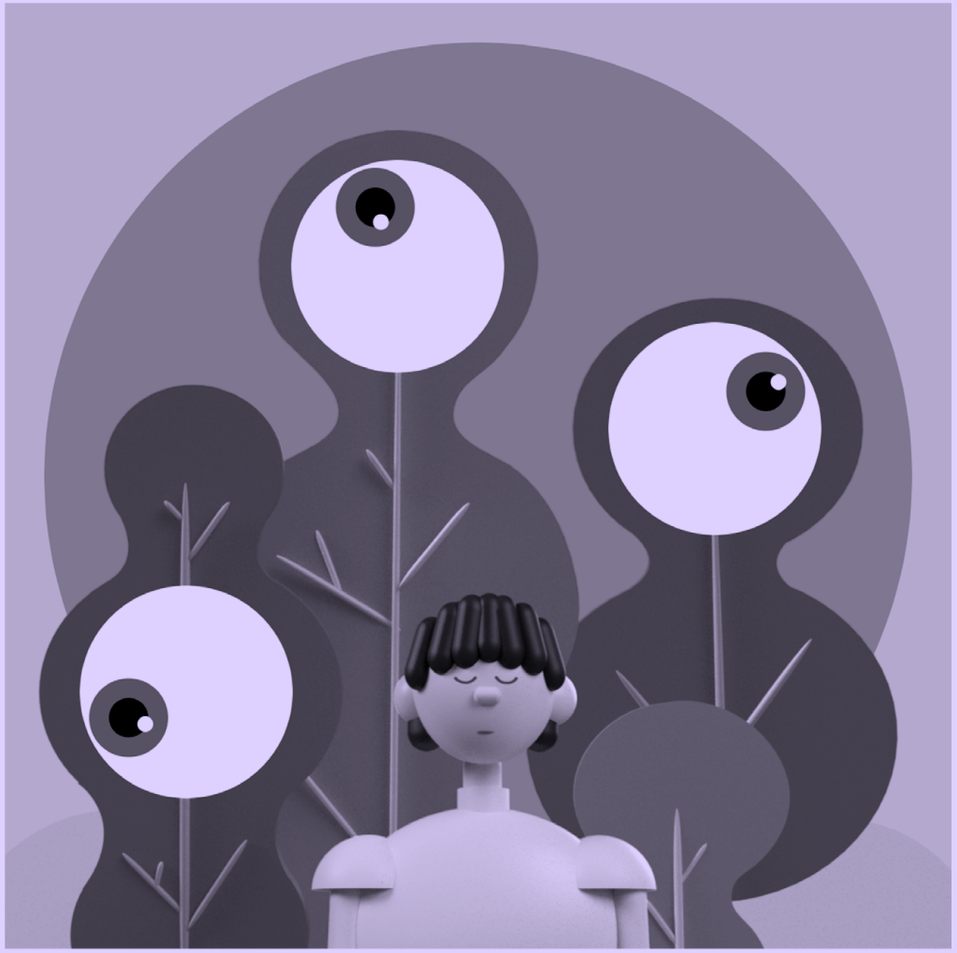
I FIRST STARTED WITH AN IDEA THAT I WANTED A DRIPPING PIPE. IT WAS IMMEDIATLY WHAT KNEW I WANTED TO DO. I SKETHED IT OUT ON PAPER BUT AS ALWAYS, THE REALITY OF MY ABILITES VS WHAT IS IN MY HEAD CAME TO PLAY. THE CONCEPT WAS THAT THERE WOULD BE A PIPE THAT DRIPS BY THE SECOND INTO A BUCKET THAT WOULD ACCUMULATE BY THE MINUTE AND A BACKGROUND THAT CHANGED EVERY HOUR. I GOT RID OF THE BUCKET BECAUSE I WAS NARROW MINDED AT THE TIME AND THOUGH I COULD ONLY MAKE IT WITH LINES AND NOT BE ABLE TO CHANGE THE FILL BUT NOW THINKING I SHOULD'VE USED QUAD . WHAT I ENDED UP WITH IS NOT TOO FAR FROM MY ORIGINAL GOAL: I STILL HAVE A PIPE BUT THE DRIP MOVES BY THE SECOND (INSTEAD OF FALLING COMPLETELY BY THE SECOND) INTO A PUDDLE THAT ACCUMULATES BY THE MINUTE ON THE SIDEWALK.



FOR THIS PROJECT I THOUGHT THAT I COULD BE CONFIDENT AT FIRST IN TERMS OF THE EXECUTION OF THE CODE. MY BACKGROUND IN CODING AND GETTING ACCLAMATED TO THIS CLASS'S PROJECTS MADE ME FELL LIKE COULD BROADEN MY CREATIVITY WITH IF AND IF/ELSE STATEMENTS SINCE I'D USED THEM SO OFTEN BEFORE. OVERALL, MY LOGIC WAS ACTUALLY MUCH BETTER, IT DIDN'T GET AS MANY ERRORS AS I USUALLY DO WHEN I'M CODING. ALTHOUGH I STILL AM HAVING PROBLEMS UNDERSTANDING THE MAP() FUNCTION, THE PROBLEM MOSTLY LIED IN ME GETTING THE SKETCH DO MY MAIN AIM WHICH WAS GETTING THE DROP FROM THE PIPE ONTO THE SIDEWALK IN ONE SECOND. I KNOW THAT FUNCTIONIS SUPPOSED TO COVERT VALUES INTO A DIFFERENT RANGE BUT NO MATTER WHERE I WOULD PUT MY "DROP" VARIABLE OR CHANGE THE RANGES IN MAP, I COULDN'T GET THE DROP TO PERFORM PROPERLY. THE CLOSEST I GOT WAS USING MILLISECOND() WITH 30000 INTO IT, BUT IT JUST MADE THE DROP FALL ONCE, OUT OF FRAME, AND NOT REPEAT IN THE SCKETCH AT ALL. THIS IS PROBABLY MY FAVORITE OF THE PROJECTS SO FAR THOUGH.

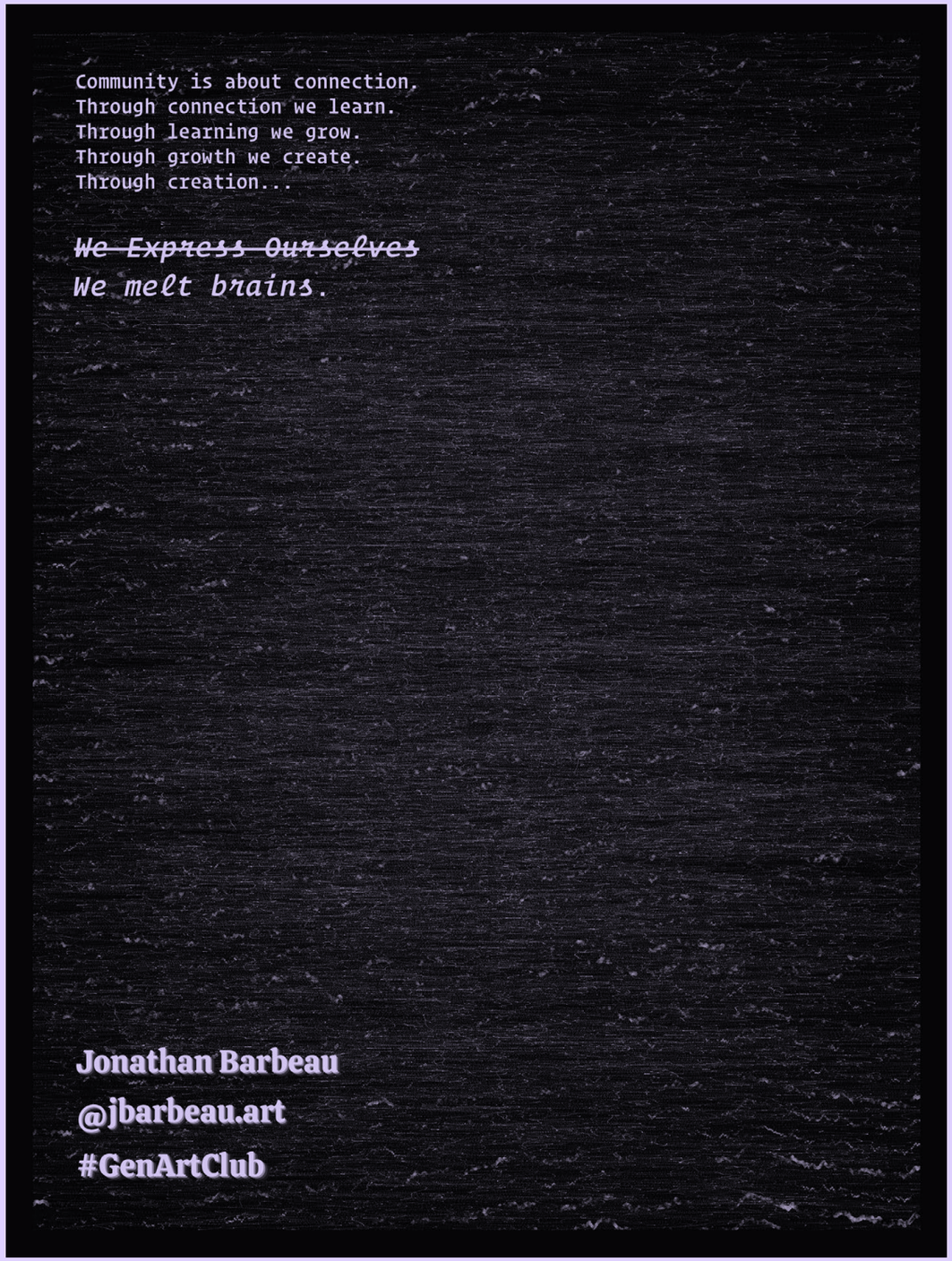
**An Experimental Clock | p5.js**
Mahima Jain | 09.2021

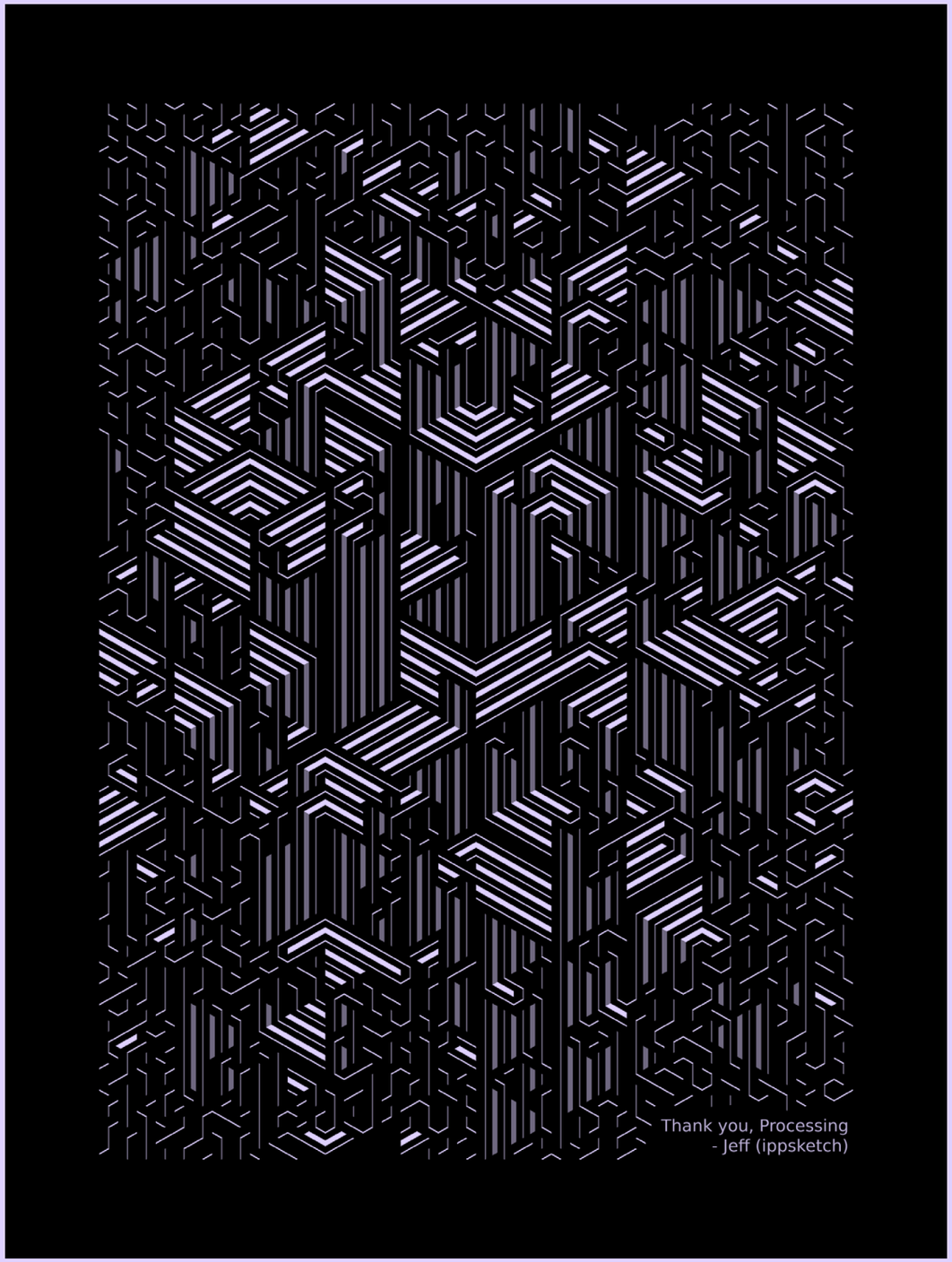
# HOPEFUL



This project is a representation of me trying to believe in nature and allowing things to happen naturally. Not worrying about shaping things, or controlling something that doesn't want to be controlled. I enhanced the static illustration by coding the 2D eyes over it and each eye ball's position depicts time in hours, minutes and seconds respectively, from left to right. I was able to bring my illustration to life and add another depth of meaning with the dynamism that javascript allowed.









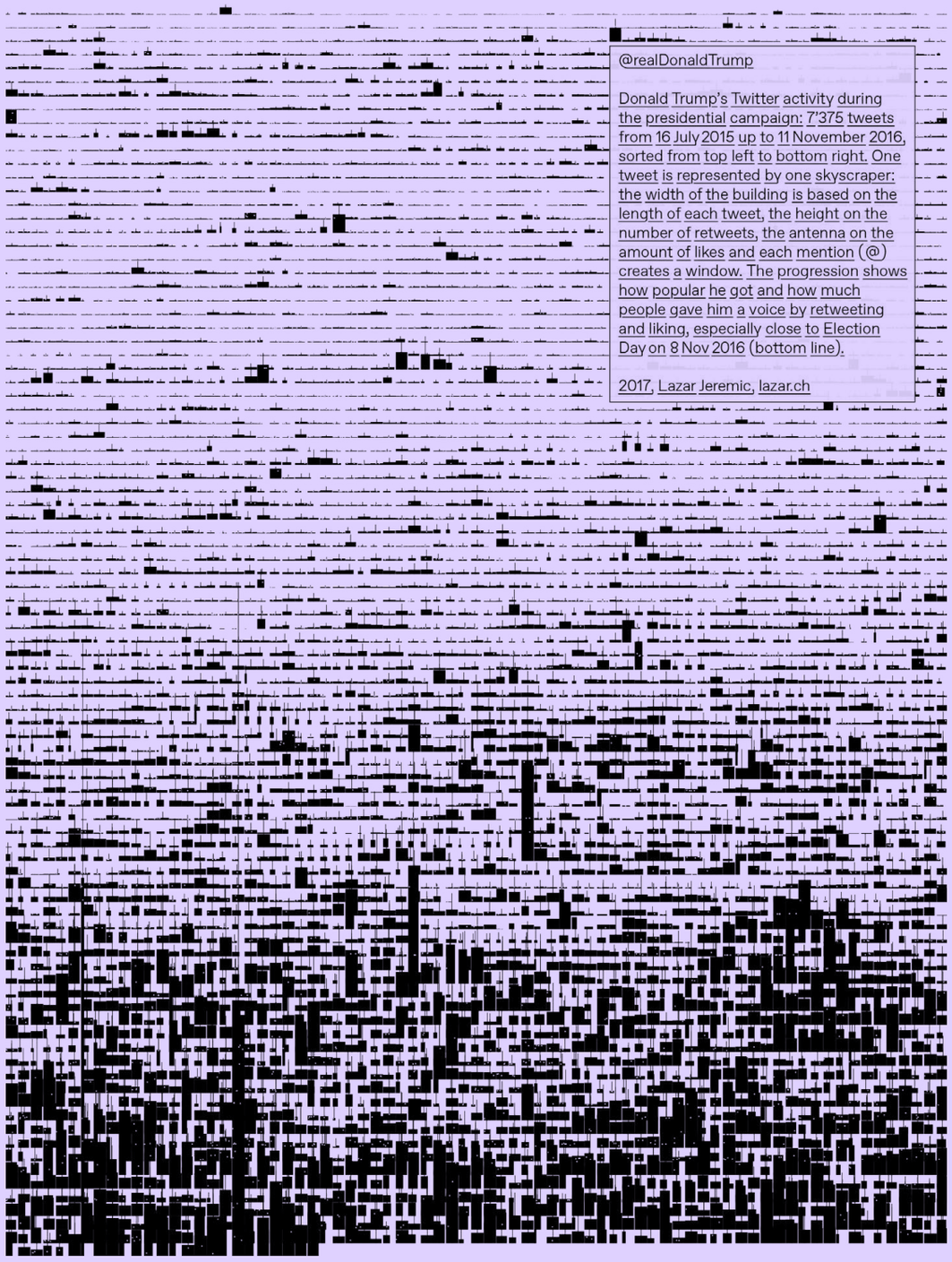


(Each word is used as a seed to create a new planet)

Me opening Proce55ing Alpha 048
in 2003 for the first time ... and ever since. <3

XOXO fjen
hen.link/fjenett



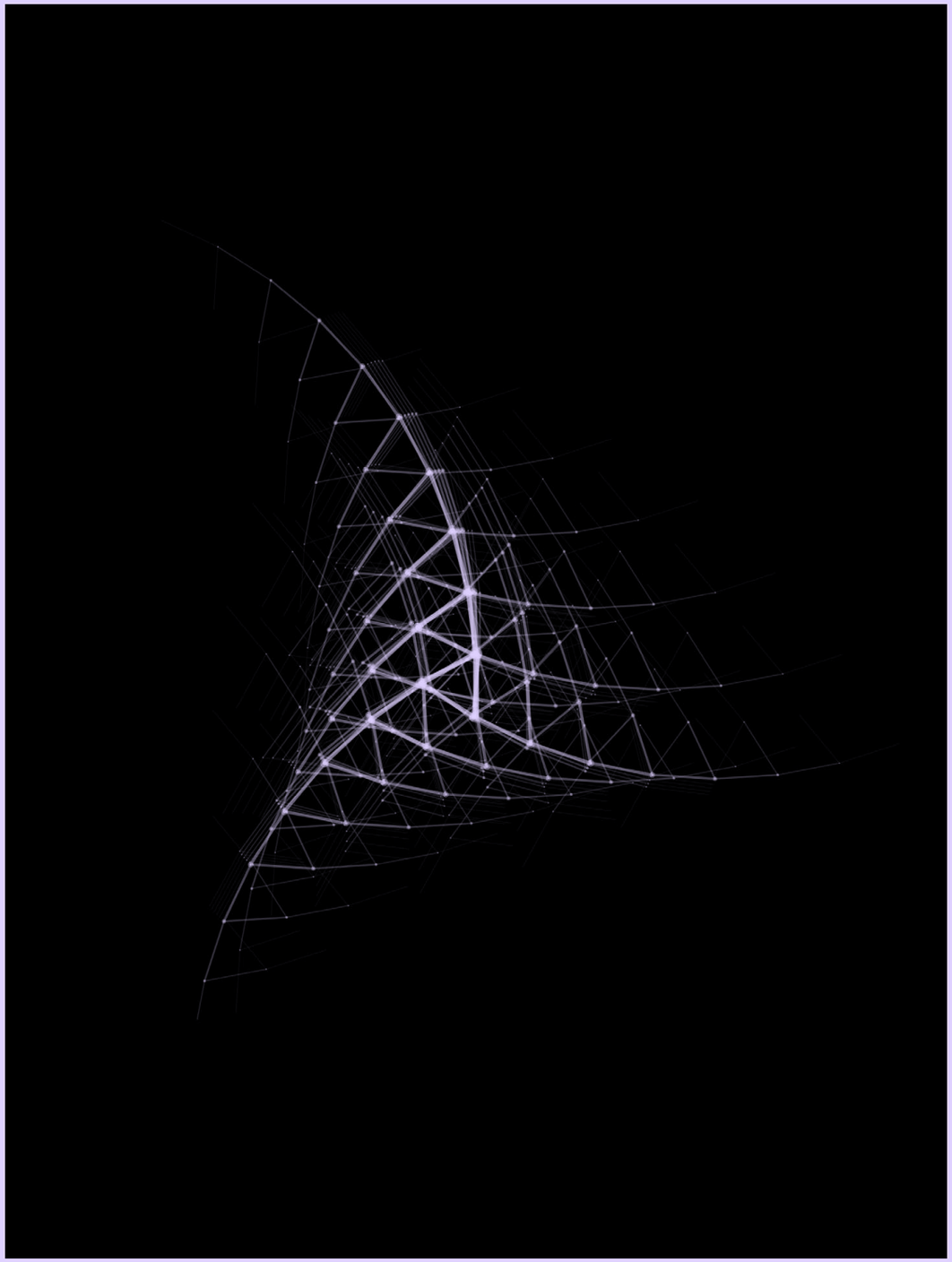



Above is a series of drawings I made for Processing Community Day 2019. These drawings were distributed as signs for the themed tracks at the event. The track names were “Under the Silicon, the Beach!” (top-left), “Accessibility, Disability, and Care” (top-right), “Radical Pedagogy” (bottom left), and “Epic Play!” (bottom right).

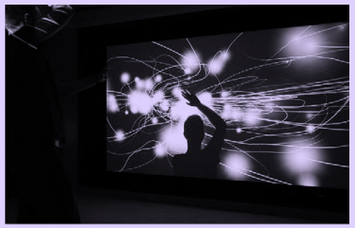
I came up with the idea of incorporating images of hands and computers after researching logos related to disability. By turning the hands into icons, they become more open to interpretation than in their functional purpose in a programming context. I want to emphasize this hands-on DIY or creative energy of the event, and I think hands are a really good indication of that.

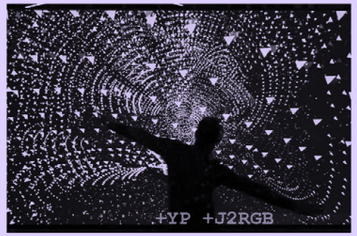


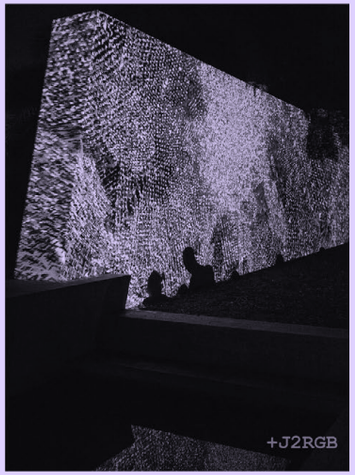


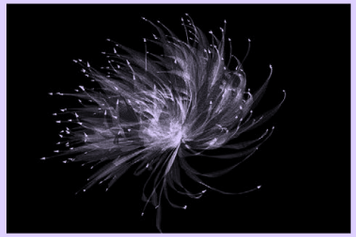


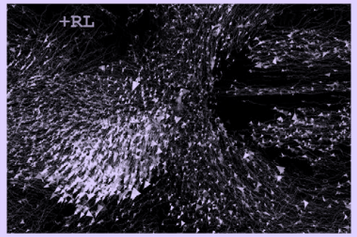


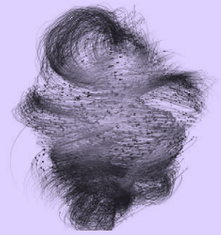


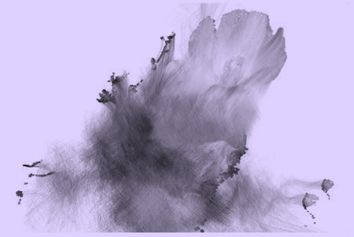



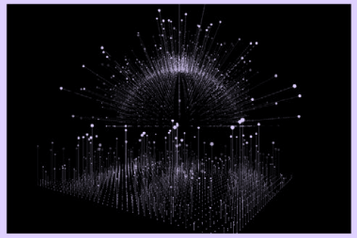



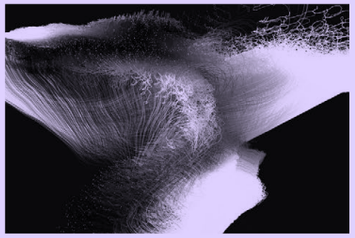



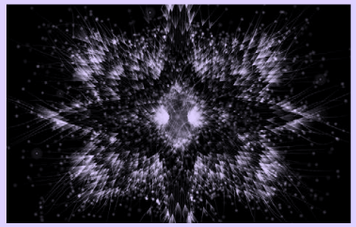


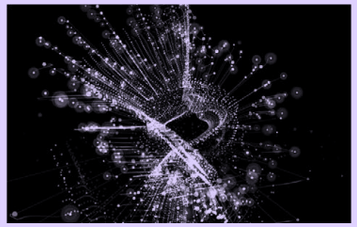











**TY PROCESSING FOUNDATION + REAS/FRY/MCWILLIAMS/MCCARTHY**
@JOHNBCARPENTER / UCLA MFA9
www.johnbcarpenter.com

J2RGB = JUSTIN SHRAKE
YP = YVES PEITZNER
RL = RUSTY LANSFORD

> Hi all,
>
> We are seeking a graphic designer to design the identity and website for p5.js, a new JavaScript library for creating graphic and interactive experiences, based on the core principles of Processing. The project will involve creating a logo and identity for p5.js, as well as working with us to design the website that will share information, documentation, and links for the project.
>
> The site will be closely linked with processing.org as we are working to maintain a connection between the two communities, but it will have its own visual style and design language.

**Jerel Johnson**
Identity Design for p5.js, website, and initial UI designer for the online editor.

I have a confession to make. I find working with others whether on a team or in a band or for a school or an organization or a company energizing, but I struggle with feelings of belonging. When Lauren McCarthy shared the call for entries for this Catalog with me I was reminded of how committed to holding space for each other the Processing Community is and has been. I wanted to say **Thank You** to the community for being generous and open. My gratitude is for everyone. There is space with you and for you because of you. All the feels.

Thanks to Reas and Fry's 2007 **Processing: A Programming Handbook for Visual Designers and Artists** I quickly moved from casual learner to someone using Processing in school a few years later; but, it wasn't until Spring Semester 2014 when I answered the email (above) that I began to contribute to the community as a designer and, adjacently, as a Creative Coding instructor. So, in what remains here, I'll add a few early p5.js design recollections:

* It was important from the start to express a situated, embodied perspective in the visual identity for p5.js as it related to the goals of p5.js's creators. I felt we all connected around a shared understanding of p5.js coming from a particular yet polyvocal place and that there was a tremendous power in that.
* It took a good number of color studies to tune our pink with the Processing Org identity while also being respectful to the brand identity of other creative coding frameworks.
* I instantly loved the asterisk symbol for the p5.js identity. I took to its capacity to express p5.js as a multiplier with verve. It gave us the **foo** of Processing times the **bar** of JavaScript taglines.
* I'm confident this was the first time I had a meme in a branding presentation (and it wasn't the last).

**Early Identity development notes**

To start, we take the name as it is:

p5.js

We see the power of Processing ideas multiplied by the reach of JavaScript by introducing the asterisk symbol:

p5*.js

And, further emphasize the ideas of Processing by removing the **j** dot:

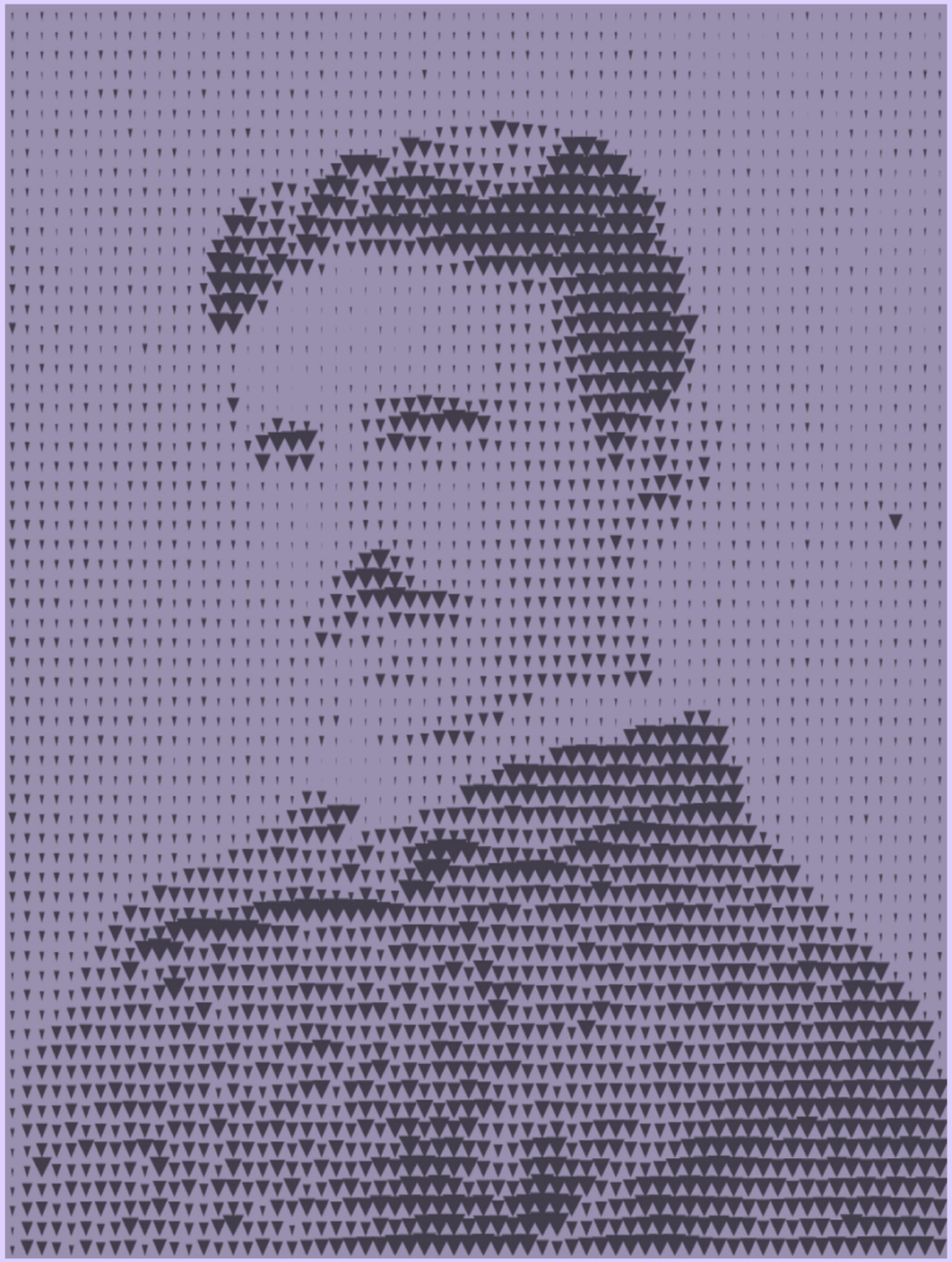
p5*.js

Finally, we adjust and redraw for openness and formal resonance:

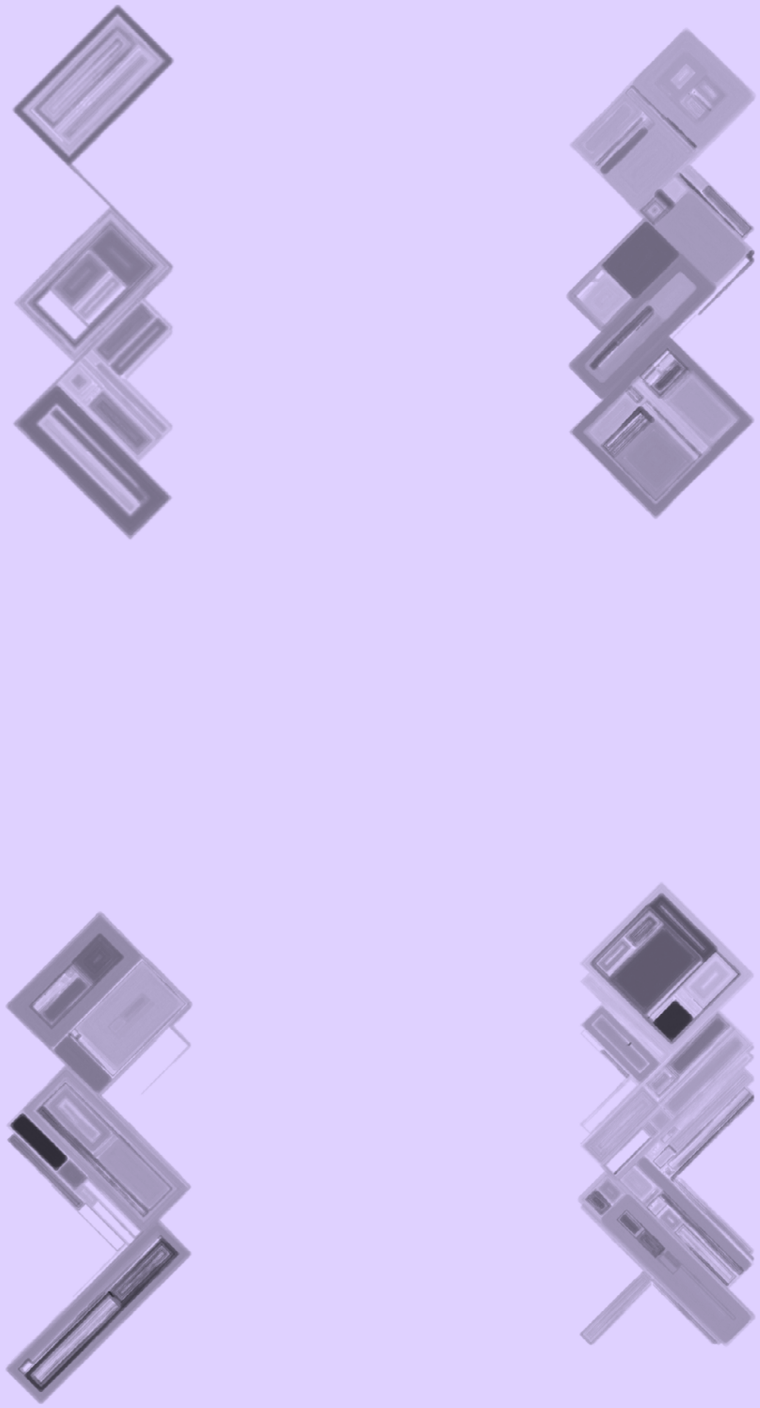
p5*.js

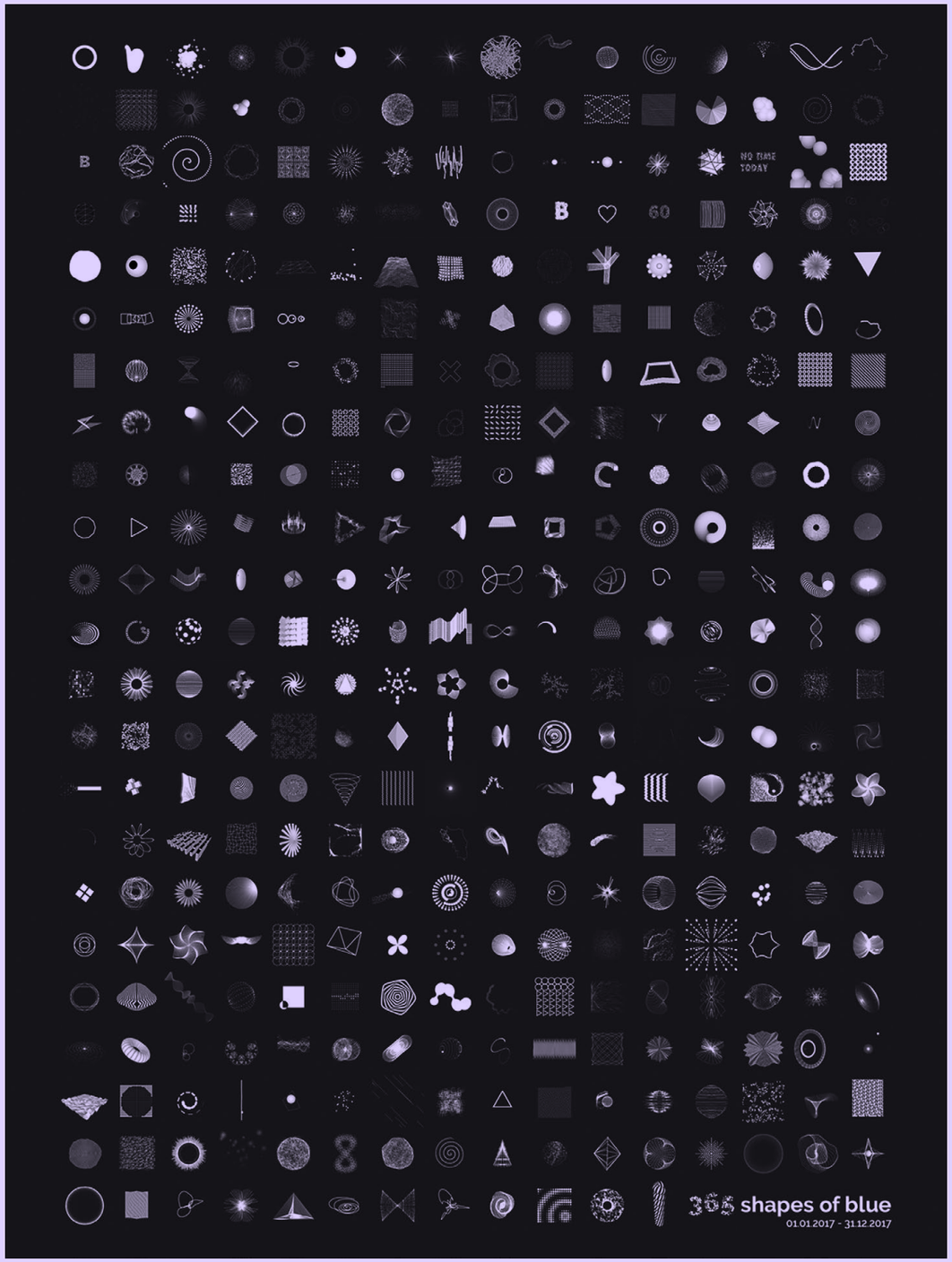
Wow. Such taglines.



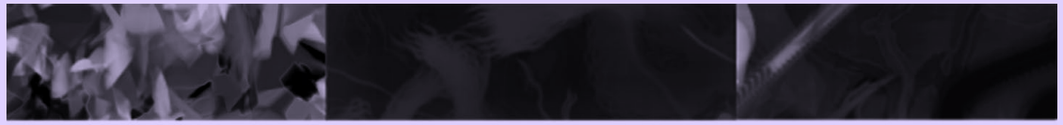










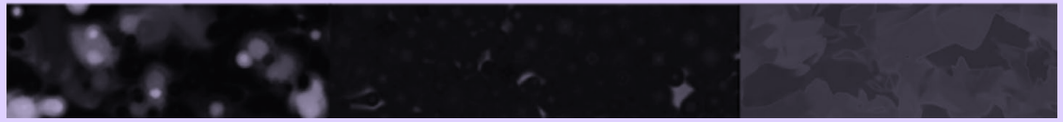



# Title:
# Society

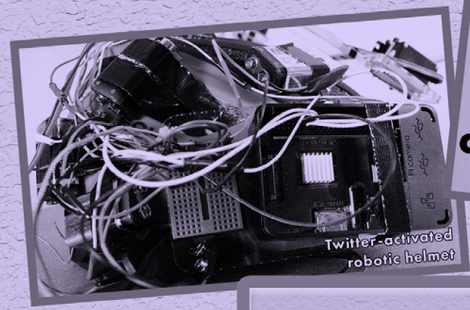

One of my college projects, the project brief is to create an ecosystem (Reference from 'nature of code' exercise).

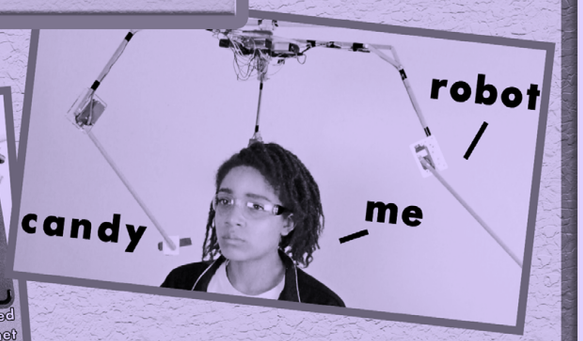
I see society as an ecosystem because humans and humans interact with each other, sometimes we help each other, sometimes we having haters.

In this project, I simulate 3 types of human in an abstract way, Human bot (person who work forever), Dreamer (wanted to chase their dream) and Killer (bad people that keep destroying others)

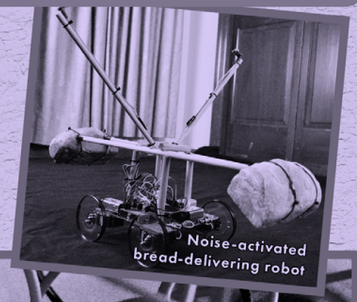


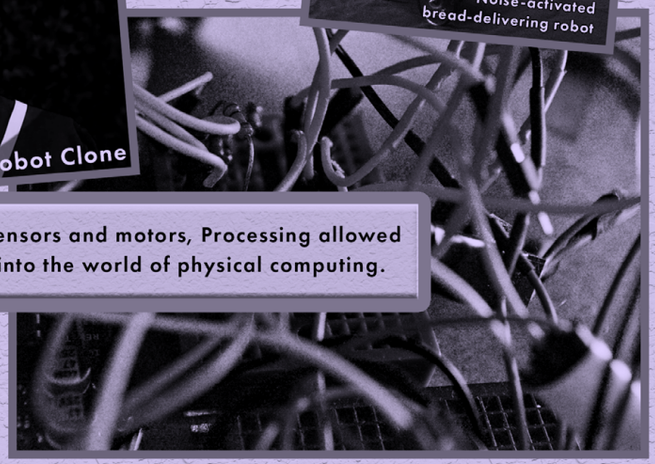
One of my earliest experiences coding was with Processing.



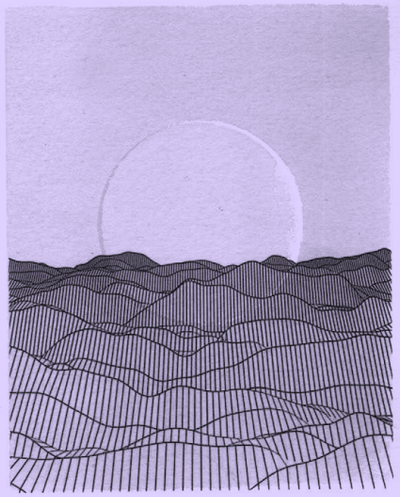

Before, I had tinkered around with microcontrollers and made LEDs blink, but through Processing, I was able to first see my thoughts articulated through code. I was able to learn the building blocks of programming, and with those blocks, I was able to build a connection between software and hardware.

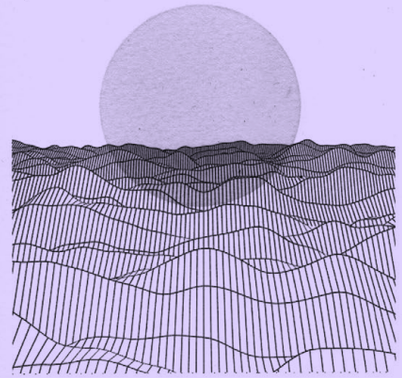



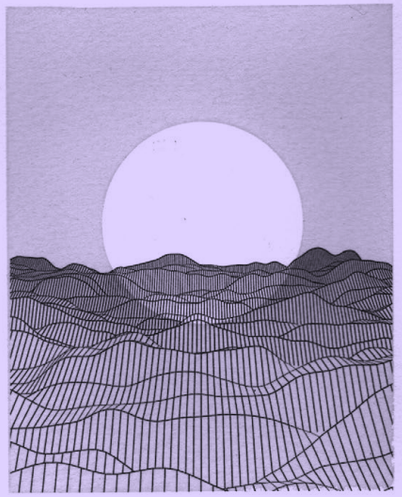



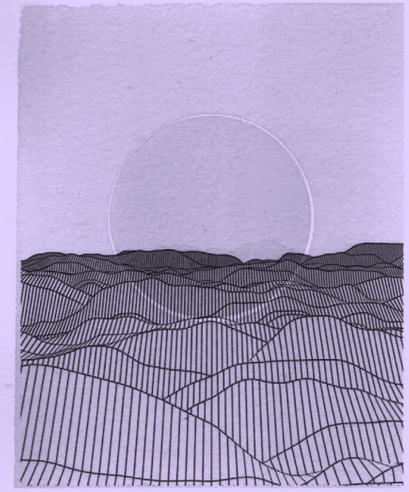
From 'Hello, World!' to sensors and motors, Processing allowed me to take my first steps into the world of physical computing.



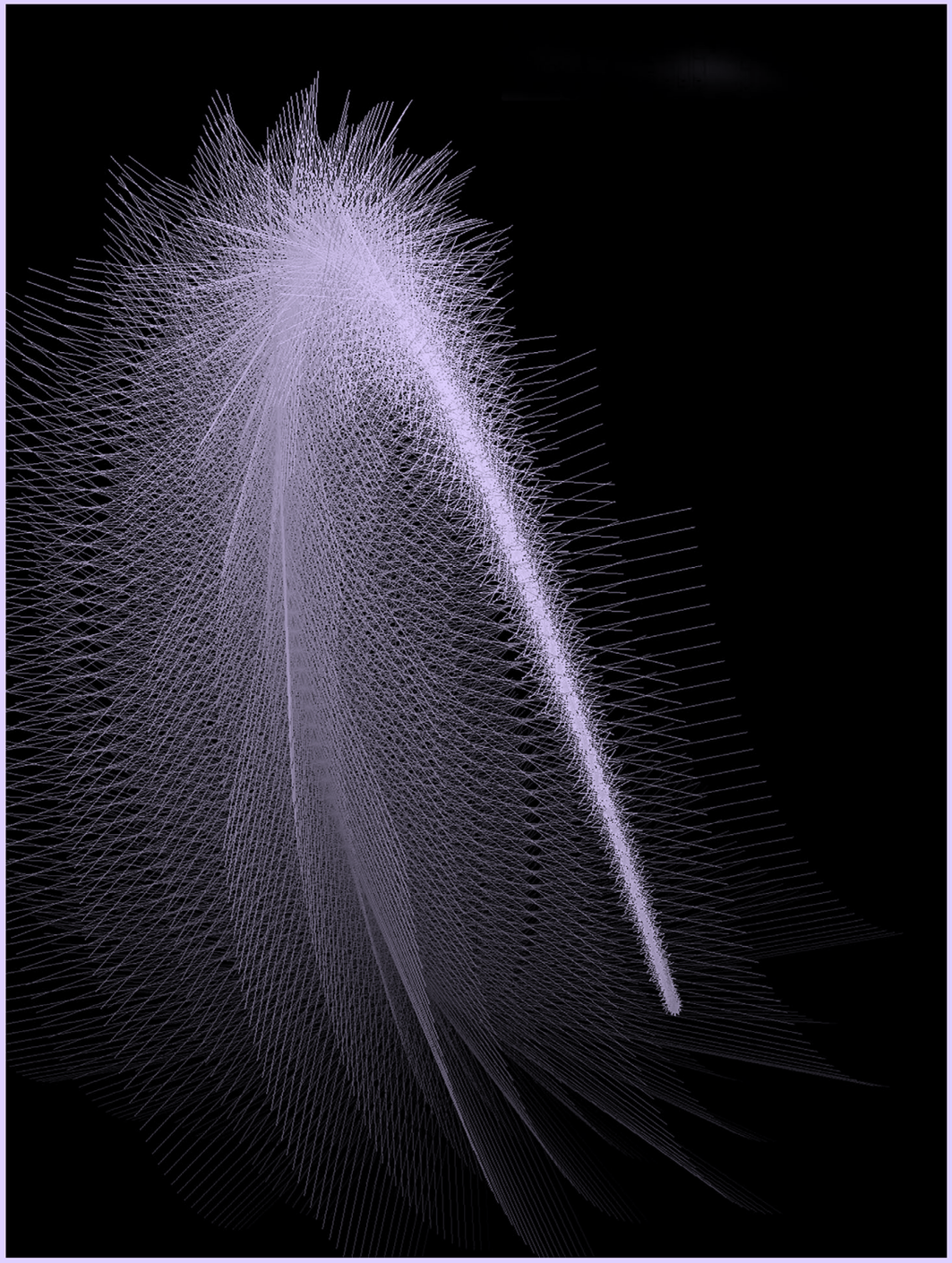






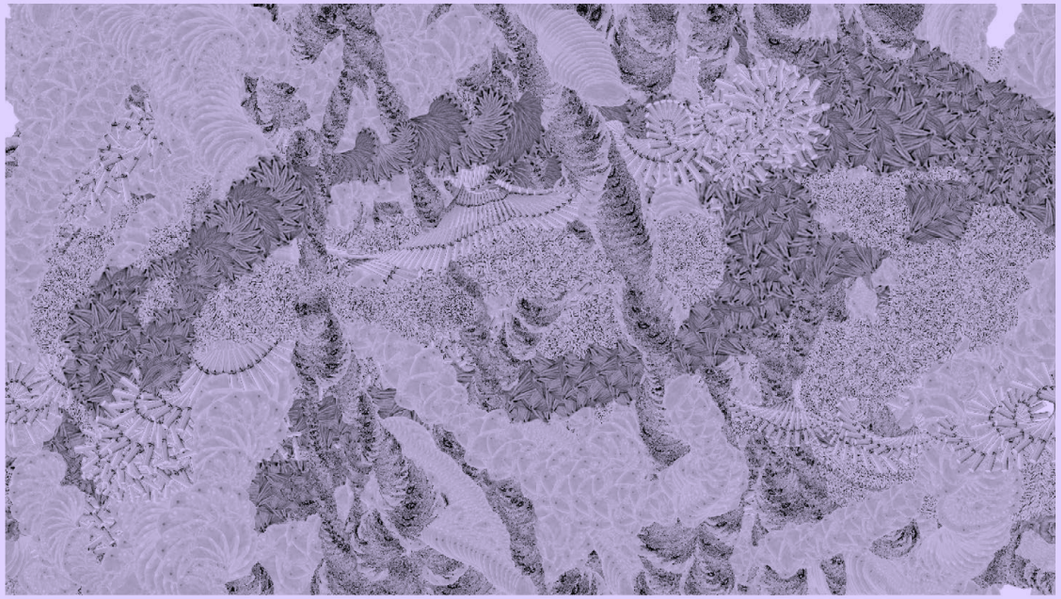






## Volcano Modak

In Indian mythology modak is considered
to be an offering to The God Ganesha.
I consider Nature as God
and for the love of food ,various ingredients
used in the making were explored.
An eruption of sweet flavors is
experienced once its popped inside the mouth.



Medium : Pixels
Tools : Processing , Rhino + Grasshopper, Spark AR ,Adobe Illustrator
Objective : Algorithmic Exploration of Form ,Pattern



# The Past, Present, and Future of p5.js Showcase

showcase.p5js.org

As a late millennial teen, my pastime was tweaking the HTML/CSS themes for my Tumblr blogs. As an undergrad at NYU in 2014, I learned to code in Processing, HTML, and CSS in classes taught by ITP and CS (CAS). As a Google Summer of Code (GSoC) participant mentored by Kate Hollenbach in 2019, I created p5.js Showcase to celebrate how people are using p5.js to make creative work, learning, and open source more interesting and inclusive. In 2020, Connie Liu featured a stunning array of creators around the world. In 2021, I got to advise Katie Chan, who introduced a beautiful theme, “The Love Ethic.”

As a grad student in computer science (MS 2022) and a woman of color of the Korean diaspora in STEM, I am in awe of how I keep finding ways to give back to a community that expands access to technology and creativity.

–Ashley Kang (강예은)
October 1, 2021

As an invited speaker at PCD Seoul on August 21, 2021, I presented (in Korean) this timeline of the past, present, and future of p5.js Showcase (쇼케이스) on the occasion of Processing’s 20th birthday. For future Showcases, I would like to manifest more themes, curators beyond GSoC, and translations.

Special thanks to Lauren, Kate, Evelyn, Saber, Connie, Yining, Joey, Katie, Sam, QT, Jiwon, Yinhwa, Roni, Phuong, Dae In, Q, Casey, Louise, Moon, Xin, and their UGA students.

Processing Community Day2019

2019년 1월

PCD @ LA

ITP CAMP

2019년 5월-8월

NYU ITP Camp

Google Summer of Code/Processing Foundation: Ashley Kang의 2019년 p5js.org/showcase

2020년 2월

Showcase launch @ p5.js v1.0 Release Party

2020년 5월-8월

Connie Liu의 2020년 showcase.p5js.org

2021년 5월-8월

Katie Chan의 2021년 showcase.p5js.org
(주제: The Love Ethic, 유색인 creator들)
...coming soon!

커뮤니티: 영어-한국어 번역

송예슬, 임인화, 김성현, Joseph Hong, Alm Chung, and more

Katie Chan의 2021년 Showcase

대학생: Google Summer of Code 참석

매년 3월에 call for student project proposals open @ summerofcode.withgoogle.com
(Processing Foundation의 open source software 프로젝트들)

2022년 p5.js Showcase 주제 idea?

# Dissociative Typeface

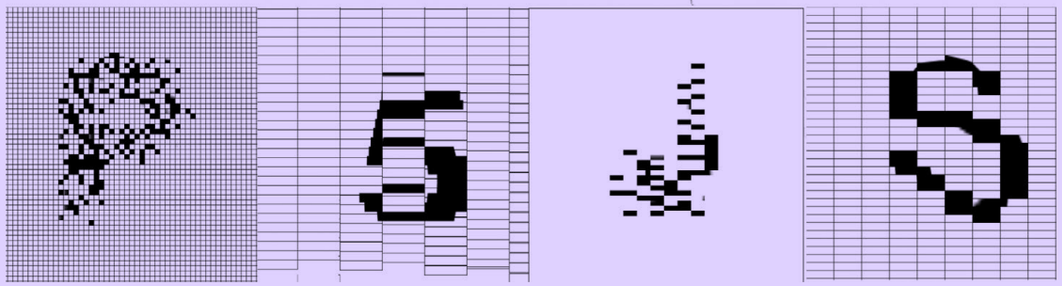
*A postmortem of familiar perceptions.*

This parametric model allows one to dissect the letters along with their grids, that we write, speak, hear and even think in.


In his essay "Art as Technique", Viktor Shklovsky put forward the concept of Defamiliarization (making strange) where he states that: "Our perception of the world has withered away; what has remained is mere recognition. It is the function of art to renew our perception. What we are familiar with, we cease to see". By making everyday objects unfamiliar, we prolong the process of perception thereby introducing a refreshing visibility, thought and curiosity in interpreting the things we have accustomed to. And nothing is more abstract than our languages, which we have accepted as granted; unquestioning the complexity of operations that relates the mental structure expressing the semantic content of utterance to its physical realization. (Chomsky).

The letters we are very familiar with become apparent with faster motion and stranger with static instances. And as Martin Heidegger remarked that it is not we who speak languages, but its the language that speaks us, this dissociative typefaces could imbibe new meanings to interpret the evolving digital cultures.

Access this model online to find at what point does a letter start becoming unfamiliar to you?





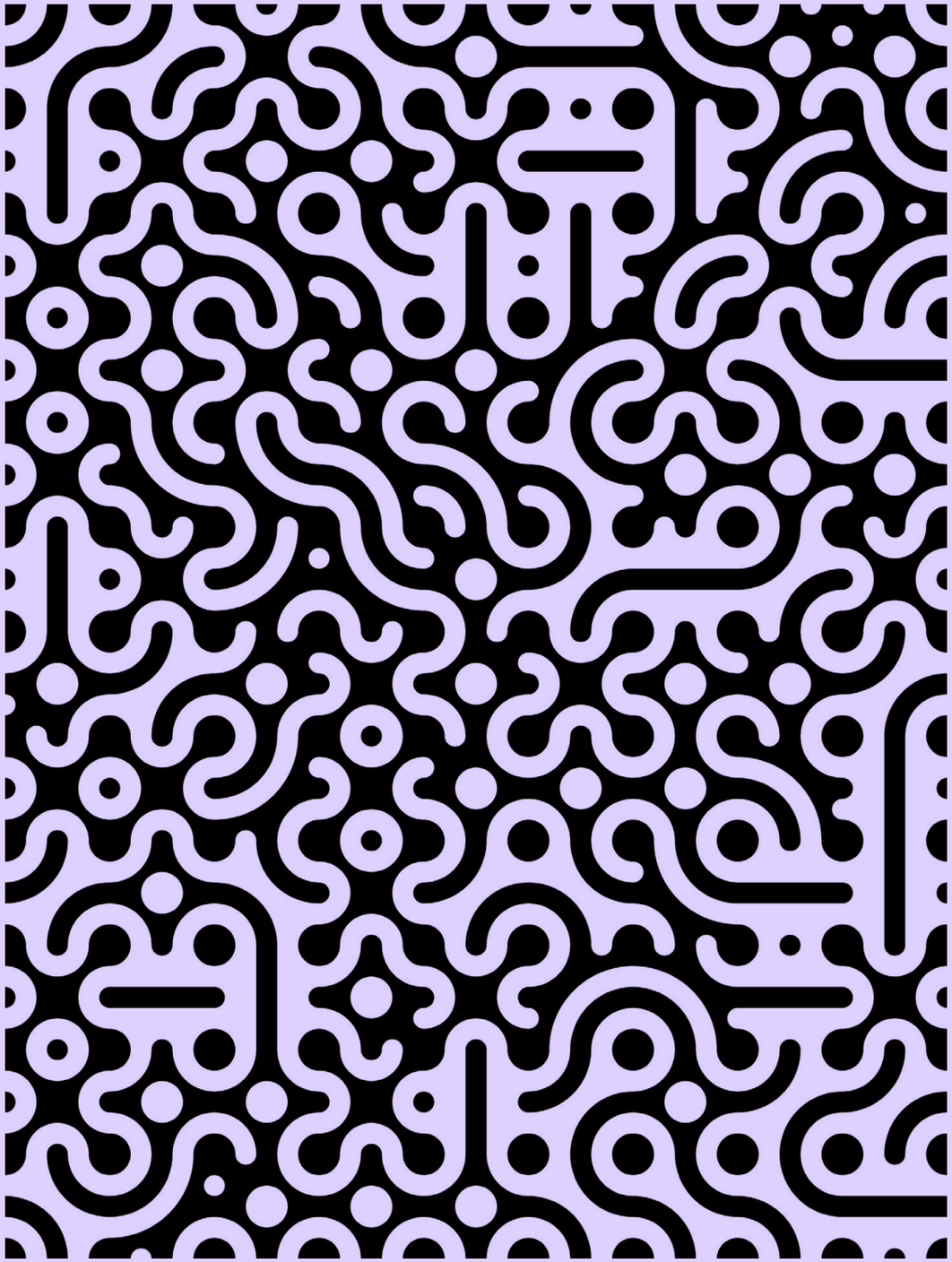

Processing is my to-go toolkit. It's a sketchbook, and I love the diversity of work you can make with it. I rendered PDFs for laser cutting, designed LED-light controllers, and taught over 100 students a year the basics of computer programming.

I'm very proud of the EyeCatcher installation. It captures my love for tinkering and experimenting with Processing. Working with alternative inputs (computer vision eye detection) and creatively rendering images on a screen, figuring out algorithms and translating them into creative output.

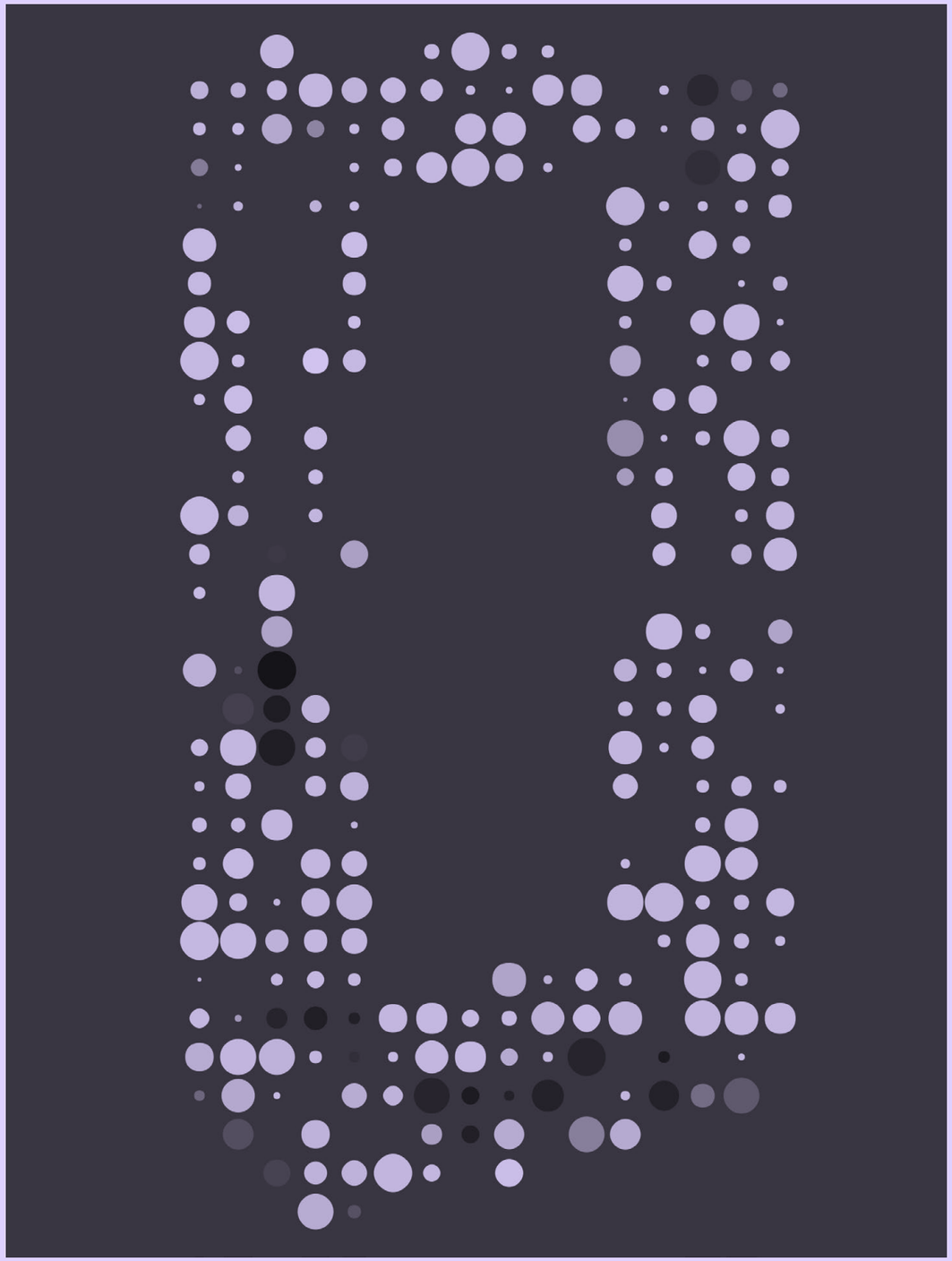
Kasper Kamperman

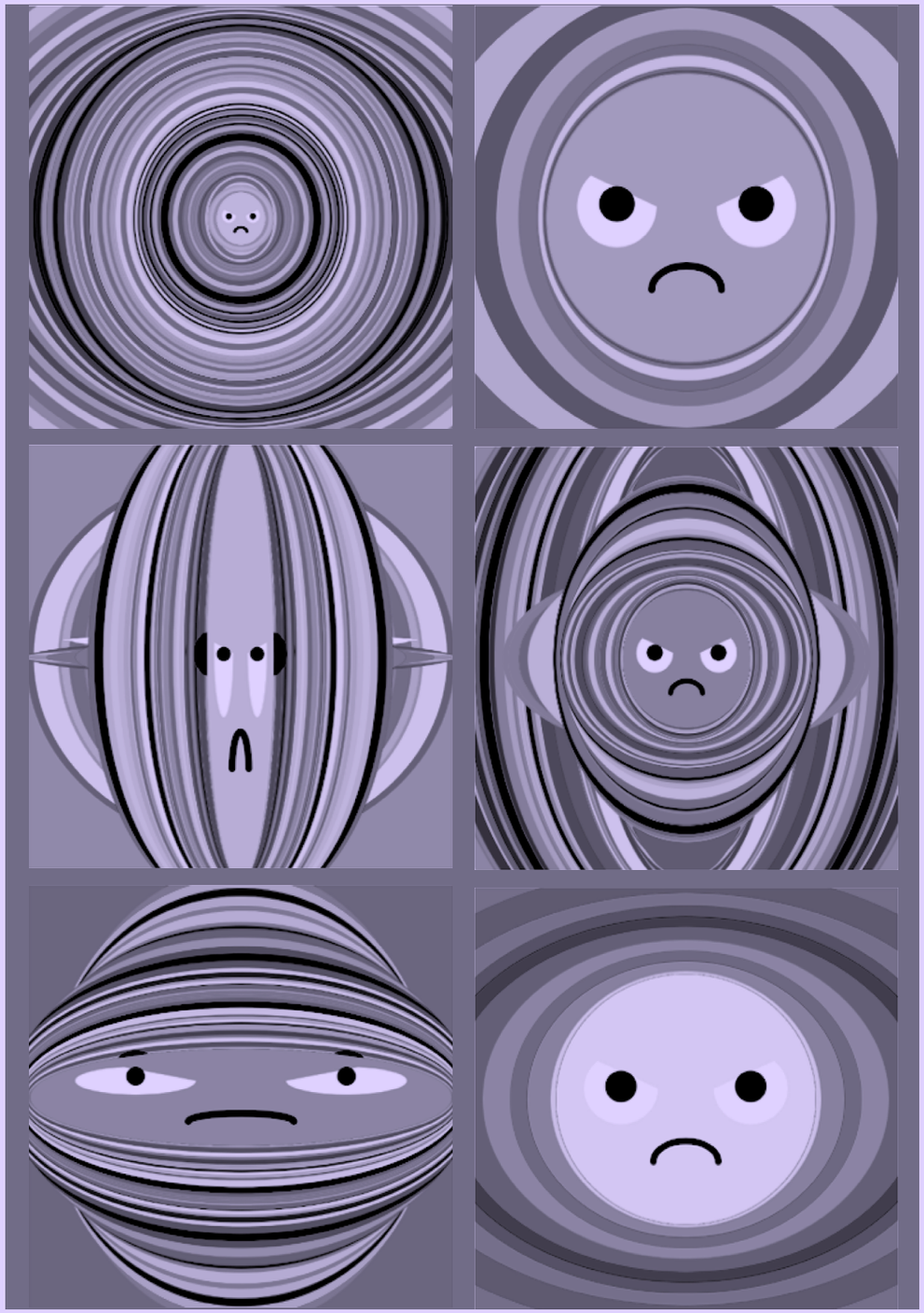
kasperkamperman.com/projects/eyecatcher/

EYE catcher

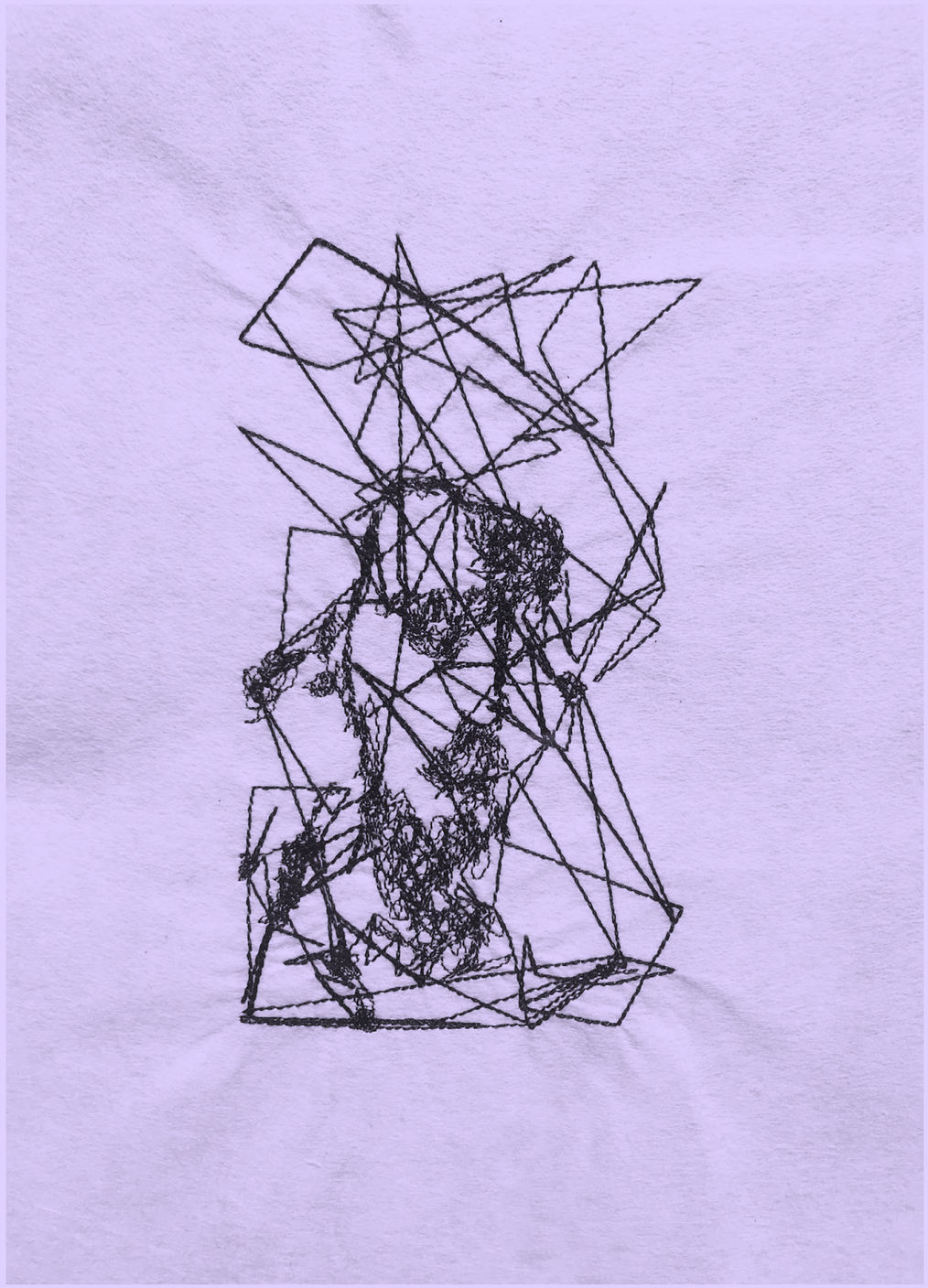


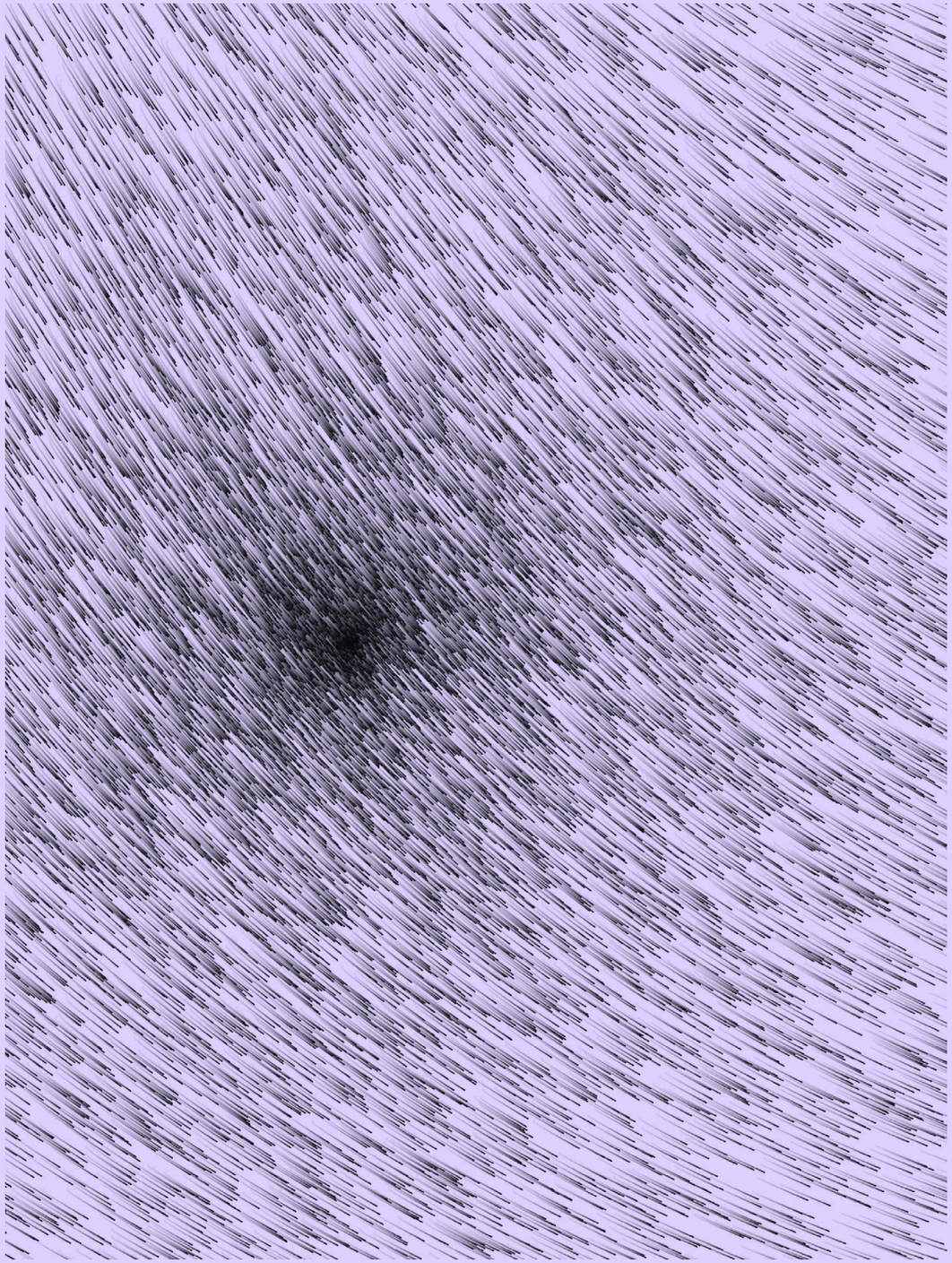




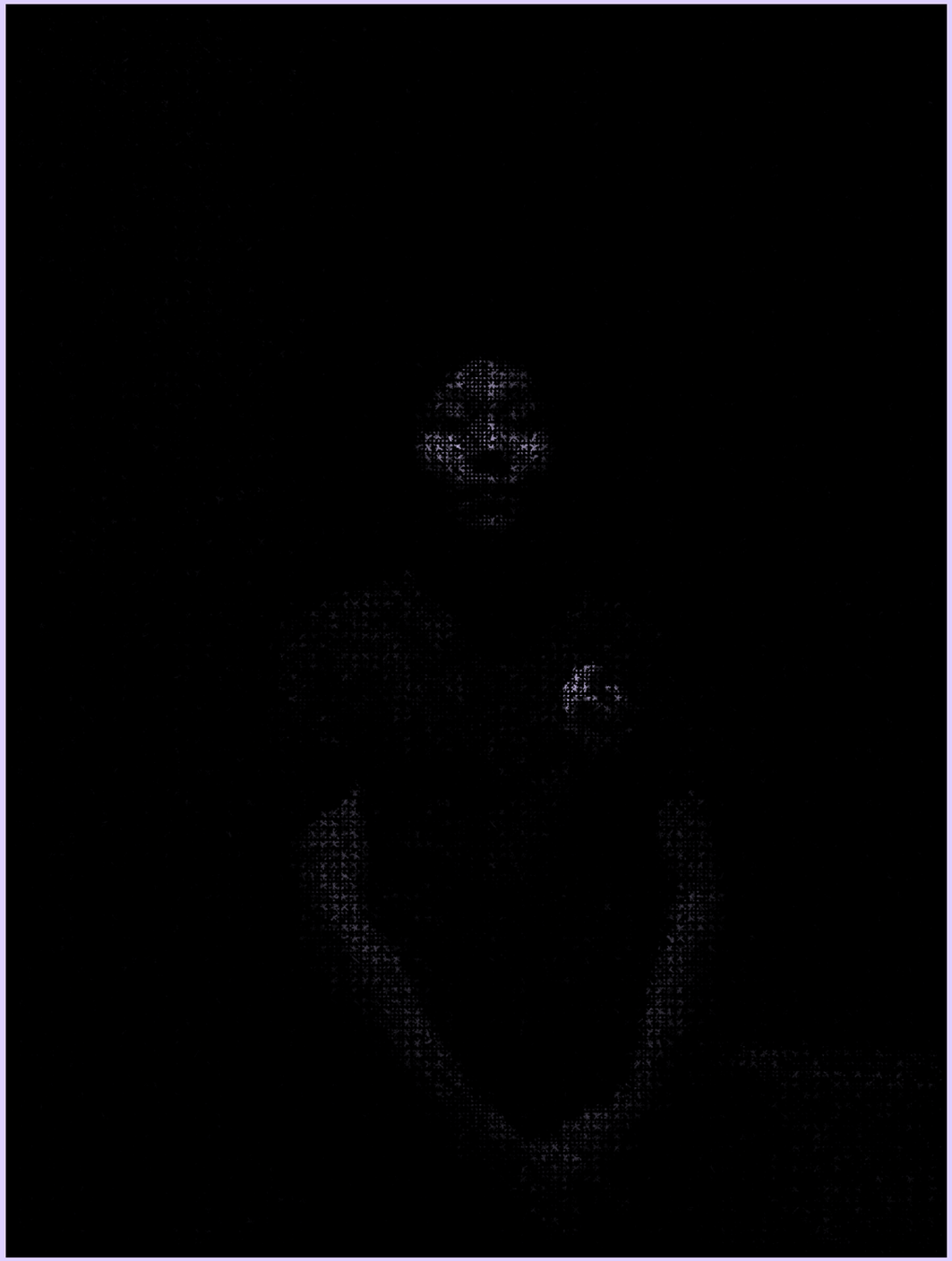






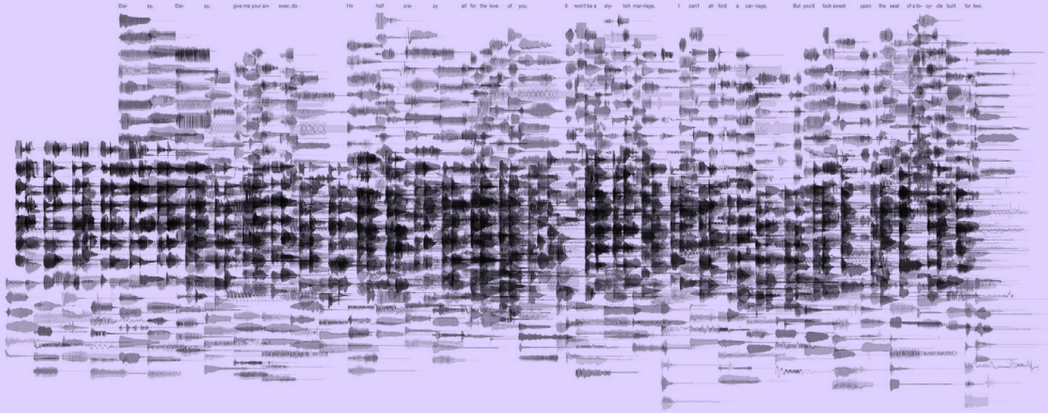


YONSAN KIM, @YONSANKM CON 279 594



KITSUNEBI, @DAI_HANABI ON TWITTER CON 280 595


AN EXTREMELY HASTILY-PUT-TOGETHER COLLECTION OF SCREENSHOTS
BANKRUPTCY
CREATIVE
ENTER YOUR NAME: ETS PLEASE





**What does being a part of the Processing community mean to you?**

I've been building custom software tools to enable unique artistic expressions, thanks to Processing and its community.

NOS

NOS is a sound reactive visual performance tool developed and used by NOS Visuals.

It is a framework written as a Processing library, that preloads multiple Processing sketches, and allows changing the active sketch in runtime.

It relies on libraries from the Processing Community for many features:

**ControlP5:** GUI elements, separate window for GUI and Canvas
**PeasyCam:** 3D Canvas with interactions for zoom, pan and rotate
**TheMidiBus:** Midi library for controllers (Korg NanoKontrol 2)
**Syphon/Spout:** Frame sharing for preview window and broadcasting to compatible mapping software
**Minim:** Interfacing with Microphone input and calculating FFT
**OscP5:** Controlling parameters via network (experimental)


Interface

**Selected Performances**

NOS Visuals, Ways of Seeing, Daejeon Museum of Art, Seoul, South Korea – 2019
Bodensee Festival – Performance with Gabriel Prokofiev, Friedrichshafen, Germany – 2018
Exploratorium – After Dark: Extended Cinemas – Performance with Amma Ateria, San Francisco, CA, USA – 2018
BINDERY: Experimental Music Night – Performance with Jonathan Crawford, San Francisco, CA, USA – 2018
Luminary Expanded – Performance with Tim Russell (Avoidance Policy), San Francisco, CA, USA – 2017
Tracks and Echoes Opening Night – Performance Matheus Coura and Keisuke Nakagoshi, San Francisco, CA, USA – 2017
Gray Area Foundation Unseen Series – Performance with Samantha Weinert, San Francisco, CA, USA – 2017
TSU! Live – Saravah, Tokyo, Japan – 2017
(Re)Constructing Nature – Performance with Amy Salsgiver Dorsay – Borusan Contemporary Museum, İstanbul, Turkey – 2016
Audiovisual Performance – Performance with Volkan Ergen – Arkaoda, Istanbul, Turkey – 2016
Midnight Orient Cafe No.5 – KC Grad, Belgrade, Serbia – 2016
Europalia Festival – Performance with Jef Neve – Flagey, Brussels, Belgium – 2016
Dogus Otomotiv New Year Celebration Party – Volkswagen Arena, Istanbul, Turkey – 2015
Signal Festival Opening Gala – Manes Gallery, Prague, Czech Republic – 2015
Turkcell Starry Nights – Harbiye Cemil Topuzlu Open Air Theatre, Istanbul, Turkey – 2015
AmberFestival Audiovisual Night – Performance with Eser Karaca – Santralistanbul, Istanbul, Turkey – 2015
International Animation Days – Performance with Eser Karaca – Institut Français, Istanbul, Turkey – 2014
Dogus Otomotiv 20th Anniversary Performance – Ciragan Palace, Istanbul, Turkey – 2014
TEDxReset – Performance with Görkem Sen, Alp Çoksoyluer – Tim Show Center, Istanbul, Turkey – 2014
Opening of Scriabin Museum – Performance with Andrei Korobeynikova – Scriabin Museum, Moscow, Russia – 2013
Deep Space Music – Performance with MaKi Namekawa – Ars Electronica, Linz, Austria – 2012

**Related Links**

https://github.com/kocosman/NOSLib/
http://nosvisuals.com/

**NOHlab**
Art Direction & Performance
https://nohlab.com/

**Osman Koc**
Creative Coder & Performance
https://www.kocosman.com/

# 비평적 기계학습을 위한 ml5.js 워크숍

2020년 프로세싱 재단과 뉴욕대 ITP의 지원을 받아 ml5.js 펠로십 활동으로 본 워크숍을 진행하였습니다.

## 발표자료

2020년 12월 2회에 걸쳐 진행된 워크숍에서 사용한 슬라이드 자료입니다.

- 워크숍 발표 자료 (1부)
- 워크숍 발표 자료 (2부)

## 워크숍 소개

알파고와 4차 산업혁명으로 많은 이들의 뇌리에 자리잡은 인공지능 기술에 대해 알아보는 실습+토론 워크숍입니다. 현대 인공지능의 근간을 이루는 기계학습 (머신러닝) 기술이 무엇인지 알아보고, 웹브라우저와 ml5.js 라이브러리를 활용해 간단한 기계학습 모델을 만들어봅니다. 기계학습이 어떻게 작동하는지, 기계학습을 활용한 응용 사례로는 어떤 것이 있는지, 이게 나와 무슨 관계가 있는지 함께 이야기해봅니다.

워크숍에 지원자격이나 필요한 사전 경험은 따로 없으며, 누구나 환영합니다. 워크숍을 준비하는 입장에서는 아래에 해당하는 참여자를 주요 대상으로 상정했습니다:

- 프로그래밍, 기계학습 관련 경험이 적거나 없는 분
- 문화예술, 시민사회 등 영역에서 활동하며 기술에 관심이 있는 분
- 인공지능과 기계학습에 대해 궁금하지만 직접 건드려보기 막막한 분
- 기술 리터러시와 교육활동에 관심 있는 분

[주의] 워크숍에서 다루지 **않는** 것의 예:

- 인공지능 전문가 되는 방법 ❌
- 머신러닝 경진대회에서 상위권 진입하는 꿀팁
- 이것만 외우면 AI 완벽 이해! 1타 핵심정리
- 현존 최고 수준 (SOTA) 어쩌구저쩌구 ❌

### 1부: 학습 Training

2020년 12월 6일, 오후 3시-5시

다루는 내용:

- 티처블 머신Teachable Machine을 이용해 기계학습 모델 훈련시키기
- ml5를 이용해 사전학습 모델 작동시켜보기
- 기계학습이 어떻게 작동하는지 함께 논의하기
- 기계학습을 사용할 때 유념할 몇 가지 질문들

### 2부: 응용 Application

2020년 12월 13일, 오후 3시-5시

다루는 내용:

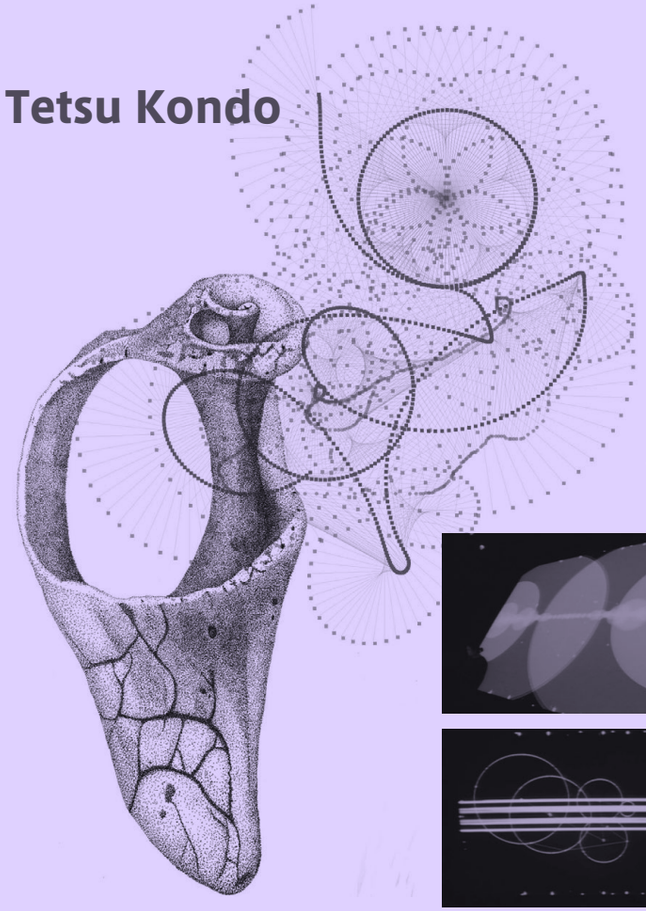
- ml5와 Teachable Machine를 함께 사용해 본인이 만든 기계학습 모델로 간단한 어플리케이션 만들기
- 기계학습 기술 인프라를 구성하는 여러 요소 알아보기
- 실생활에서 만날 수 있는 기계학습 응용 사례 살펴보기
- 기계학습 및 연관 기술과 나의 관계에 대해 생각하기

준비물: 구글 크롬 브라우저가 설치된 데스크톱 (또는 노트북 컴퓨터) 환경, 키보드, 마우스, 웹캠, 스피커 (또는 헤드폰).

워크숍에 참여하는 분들은 p5.js 커뮤니티 성명서를 읽고 오셔야 합니다.



# Tetsu Kondo

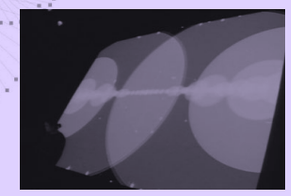


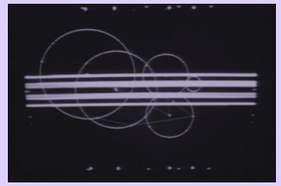
"Nothing the Same" - drawing and coding (-now)

A Japanese designer, artist, musician, creative director, and professor born in Toyokawa City. His breadth of expertise ranges from interactive design, user experience, and software development to live audio-visual programming and art installations. He has been practicing meditation and plays the guitar every day, and sometimes the didgeridoo.

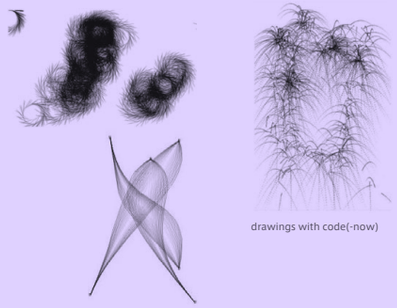
He graduated from the Interactive Telecommunications Program (ITP) at New York University (NYU) Tisch School of Arts in 2002, and became a Resident Researcher.

## with Processing





Dendraw - live audio visual software (2007)





drawings with code(-now)

He has performed and presented his art works globally at various venues such as FILE Hypersonica in Sao Paulo, Eyebeam in NYC, CENART Centro Multimedia Center in Mexico City, and Inter Communication Center (ICC) in Tokyo. He teaches media art and design in Japan.

Tetsu runs Tetraleaf, Inc., a media design company which handles various forms of production, and he is a cofounder of Field+Line, a creative partnership with Jeff Feddersen.

Installation for Tokyo Sky Tree Sumida Aqualium - F+L project with Jeff Feddersen (2017)

# Processing1.0 Alpha Reference translation in Japanese（日本語）


## The first encounter with Processing by Tetsu Kondo

My first meaningful encounter with Processing started with a short email to Casey in 2002. I had just graduated from New York University's ITP as an international student, and became a Resident Researcher under my great adviser Red Burns, mainly researching physical computing and making installations and musical instruments. During my free and unique research period at ITP, I became interested in the gentle existence of Processing. I sent an email directly to Casey and started translating the reference pages in order to study this language in depth. From this point on, my creativity slowly but gradually spread as if from a deep spring, and Processing became the starting point for creating the basic philosophy of my media art, design, and educational activities. It still has a great impact on my life. I vividly remember the beautiful encounter with Processing at that time, and it will continue forever. Thank you Processing.

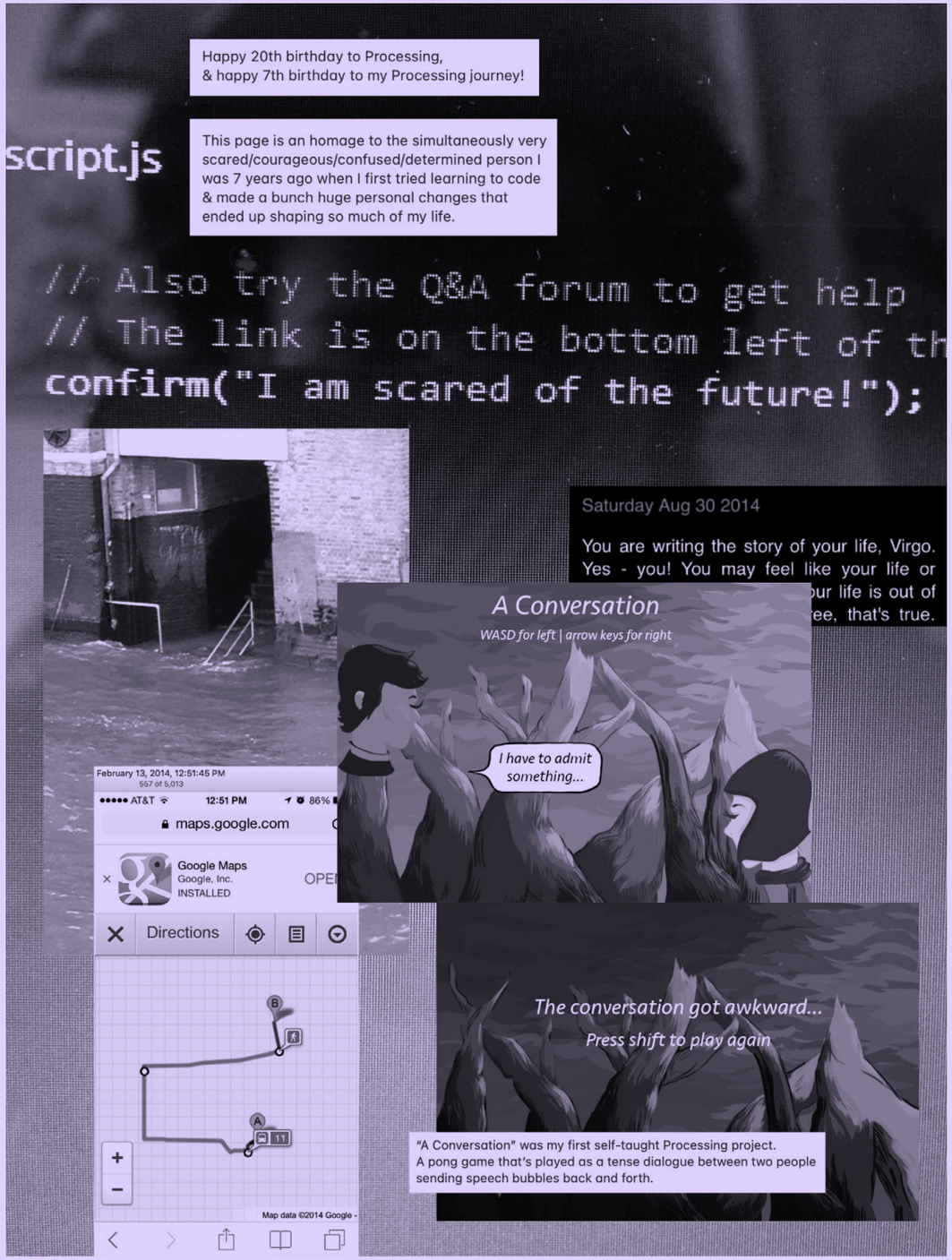
Alpha 1.0 Reference pages in Japanese (2002)

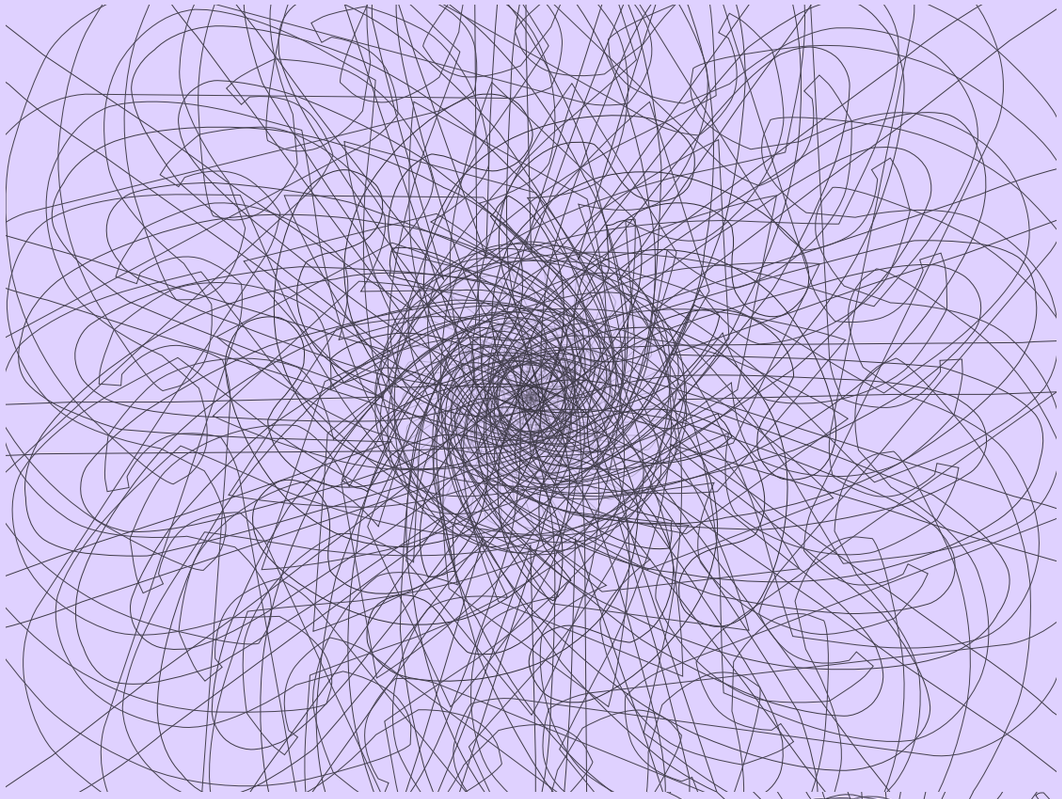
## Tetsu Kondo

A Japanese designer, artist, musician, creative director, and professor born in Toyokawa City. His breadth of expertise ranges from interactive design, user experience, and software development to live audio-visual programming and art installations. He has been practicing meditation and plays the guitar every day, and sometimes the didgeridoo.

Tetsu runs Tetraleaf, Inc., a media design company which handles various forms of production, and he is a cofounder of Field+Line, a creative partnership with Jeff Feddersen.



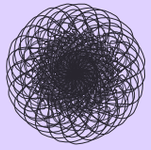




**metaGeometry**

Video installations of typographic character outlines, rotating and scaling around a central point. Debuted in the KOBRO Gallery, at the Academy of Fine Arts in Lodz, Poland, in 2011.

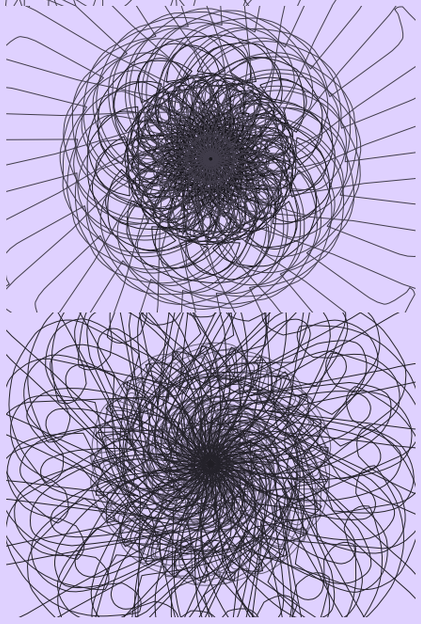
Visual and motion design by Jan Kubasiewicz; Processing code by Scott Murray.



**Jan** would like to thank Scott for translating his vision of metaGeometry concept into a visual experience, through Processing.

**Scott** would like to thank Jan for introducing him to Processing, back in 2007 at the Dynamic Media Institute.

jankuba.com
scottmurray.org





## Processing for Android – Current Progress

Processing for Android helps to create Android apps with ease, including live wallpapers, watch faces and VR apps. As the android changes rapidly, there is a scope of improvement for Android Mode. Nowadays many people use Processing for Android, but few of them know the history of it. In an article, Ben Fry (Co-Founder @ProcessingOrg) mentions that he started working on the project with Casey Reas (Co-Founder @ProcessingOrg) back in 2009. Yes, Processing for Android is more than 10 years old and still works great after these many years with the continuous efforts of the contributors including Andres Colubri (Project Maintainer- Processing for Android).

I first came to know about Processing sketches through APDE which is an IDE to create and run Processing sketches on the Android Platform. Later I started working on Processing Development Environment with Android Mode. One of the things which I like the most about Android Mode is its structure. It is interesting to know how various submodules (Core, AR, VR, and Mode) combines to form the Android Mode. It fascinated me to know more about how the various tasks like generating AndroidMode.jar and related Core, AR, and VR libraries are all builds with a single Gradle command.

I get connected to Aditya Rana (GSoC'20 Student and GSoC'21 Mentor) who introduced me to the project. I often see him encouraging me and reviewing my contributions to Processing for Android.

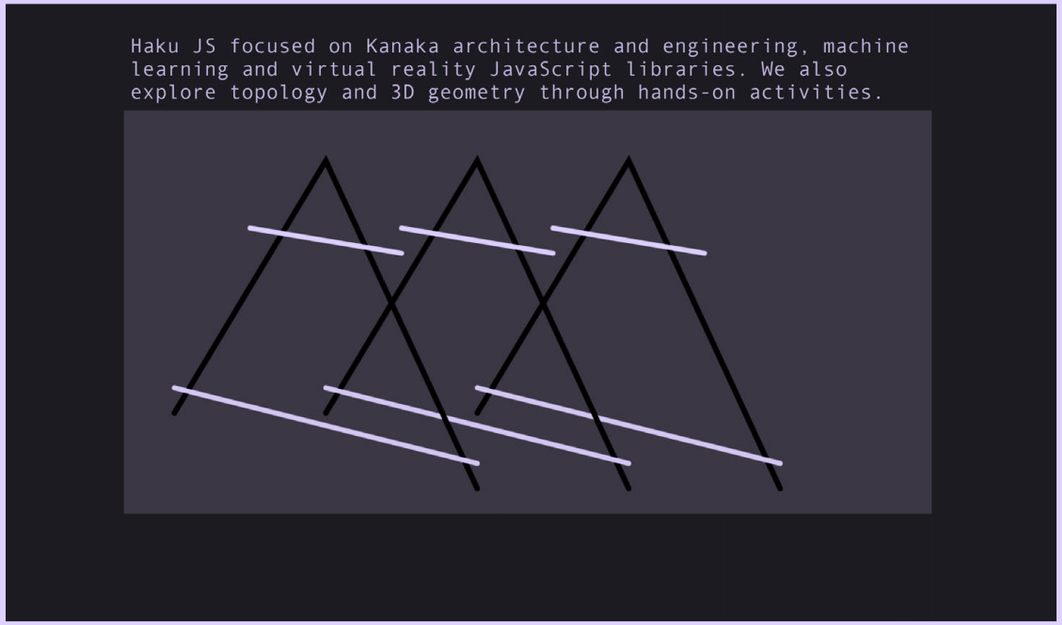
Some of the key areas on which I have worked are -

- Providing AndroidX support
- Improving permission box UI
- Website Improvements
- Signed AAB Export Support

I believe the official processing discourse platform is a great way to learn, share knowledge, and improve projects based on user experiences. There will be a scope of progress in the project in some of the areas including Java/Gradle Version Update, V2 Signature Scheme Support for packages and bundles, etc. Ideas like providing Multiplatform Kotlin Support for both Android and iOS devices show the broader aspects of the Processing for Android.

- Rupesh Kumar (iamrupesh.me)

HO'OHIWAHIWA I KĀ MĀKOU HAUMĀNA O HAKU JS, PURPLE MAI'A: THOMA, KALIKO'AULI'I, JAKE, CARYS, KA'ENA, A JACOB!



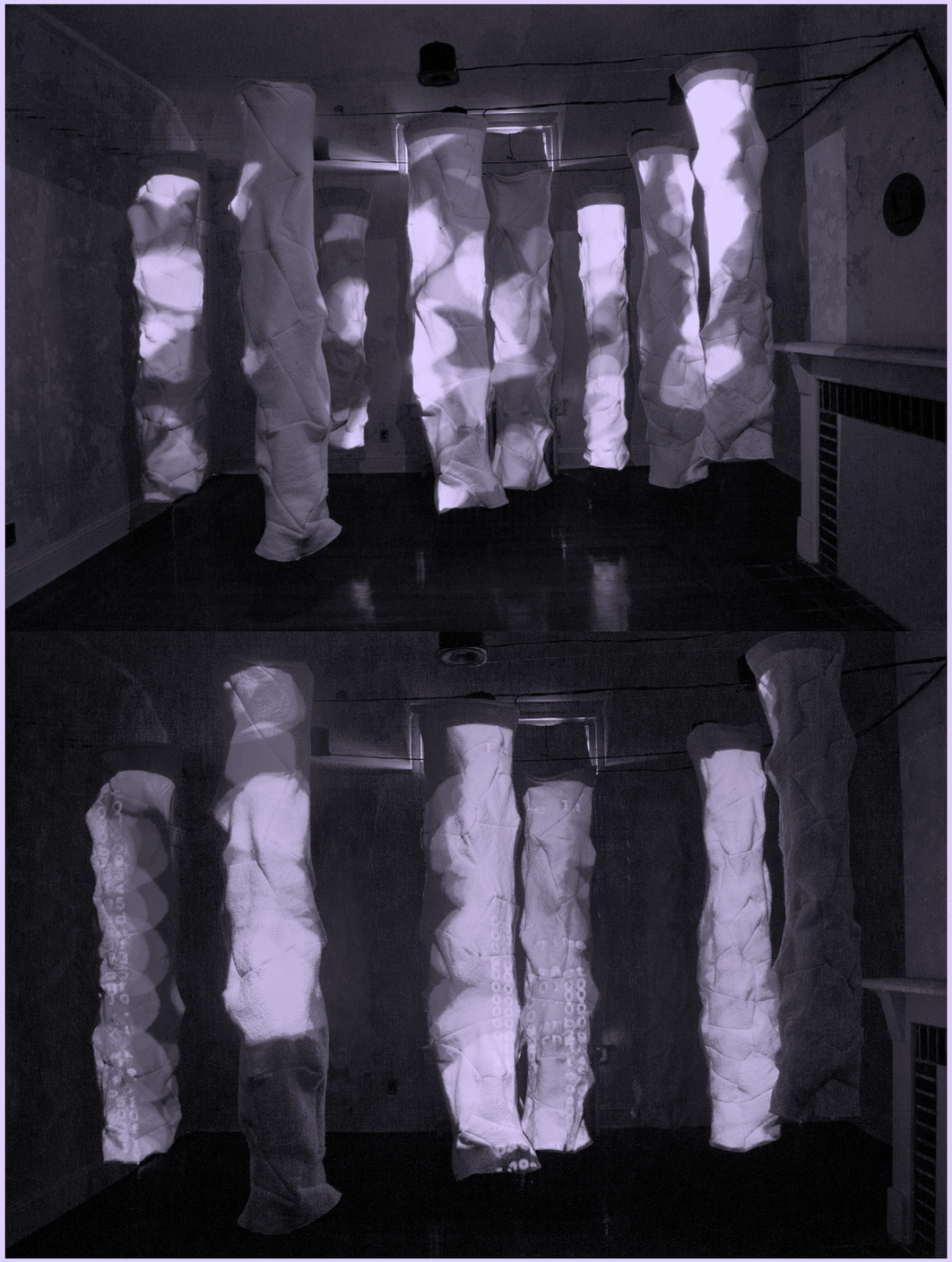

*Alt-text:A generative architectural "brown print" of a Hālau Wa'a, which is a traditional Hawaiian structure for storing outrigger canoes.*

Purple Mai'a was founded by Olin Lagon, Donavan Kealoha, and Kelsey Amos in 2013. At that time the idea was simple: start a nonprofit that teaches Hawaiian kids how to code. But we quickly realized that our hopes for our keiki and our ambitions for Hawai'i's people were much bigger than just coding. We wanted to give 'ōpio the skills, mindsets, and mālama that they need to not only succeed personally but to solve problems for their communities in the 21st century. And we wanted to fundamentally change the perception that there is anything contradictory or unusual about Kānaka Maoli and other people of Hawai'i being excellent in areas like technology and entrepreneurship.



**kusakari**

Making generative art with p5.js everyday
OpenProcessing: https://openprocessing.org/user/224308
Twitter: @kusakarism





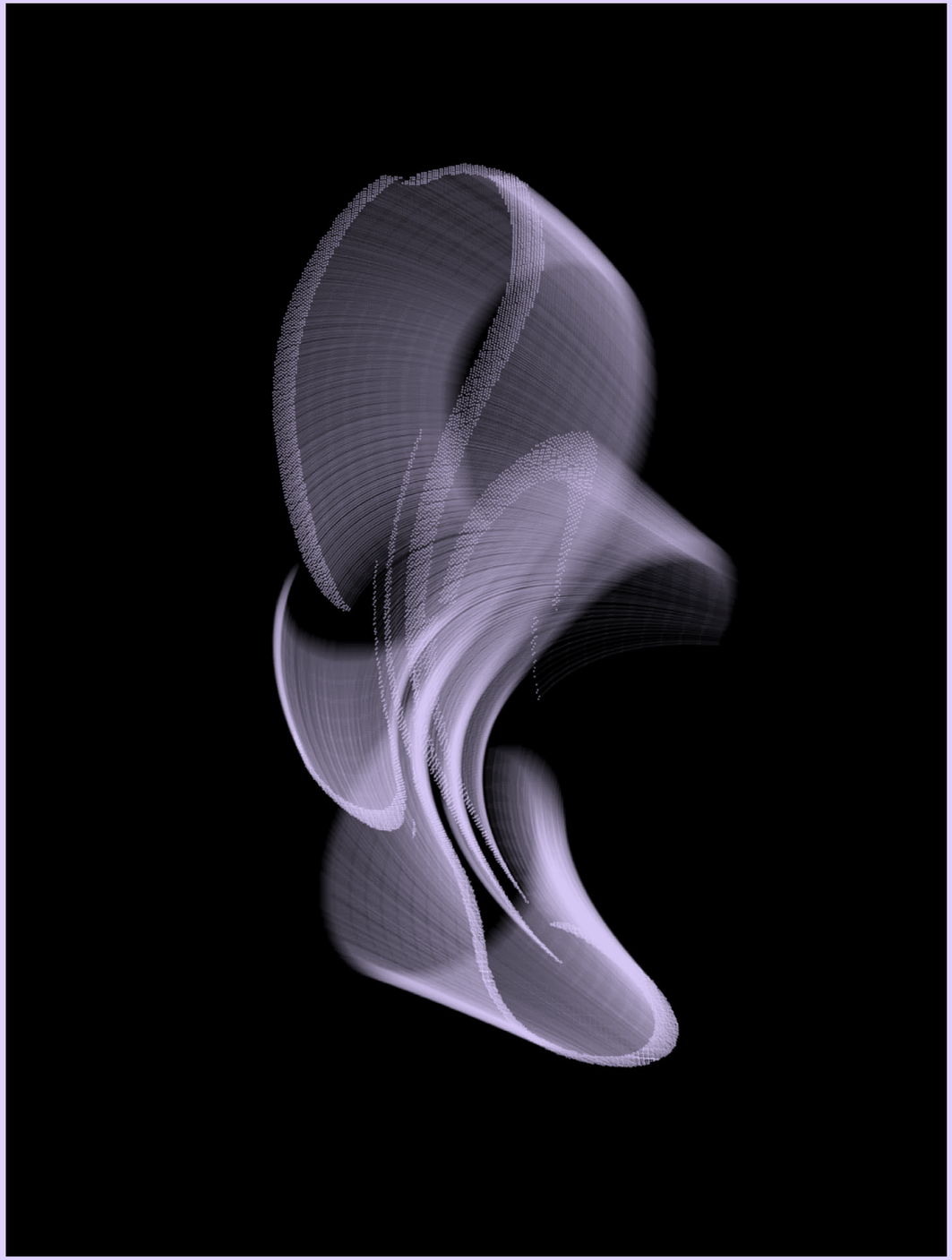

W3FI by Laleh Mehran & Chris Coleman
W3
W3
W3
W3FI
W3FI
NUMBER OF ROUTERS: 82108


SOME YEARS AGO I WENT TO THE GOOGLE SUMMER OF CODE MENTOR SUMMIT IN SUNNYVALE CALIFORNIA TO REPRESENT THE PROCESSING FOUNDATION WITH JOHANNA HEDVA

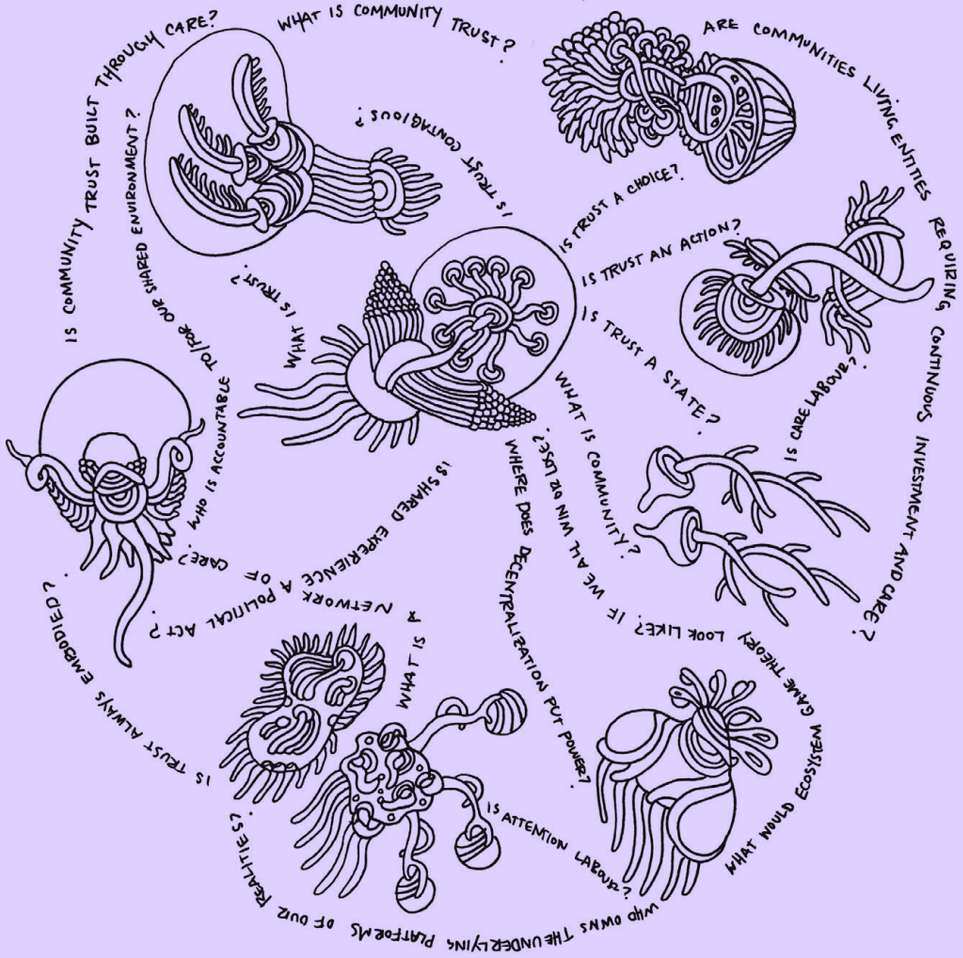
I WAS MEETING JOHANNA WHO IS NOW A LIFELONG FRIEND FOR THE FIRST TIME IN THE HOTEL BAR AND I TOLD HER TO LOOK OUT FOR SOMEONE WITH A SILLY HAT

LATER I WALKED DOWN THE BLOCK FROM THE HOTEL AND SAW THAT LOCKHEED MARTIN WAS ACROSS THE STREET FROM YAHOO AND UNDERSTOOD VISCERALLY HOW THERE WAS NO DIFFERENCE BETWEEN SILICON VALLEY AND THE WAR ON TERROR



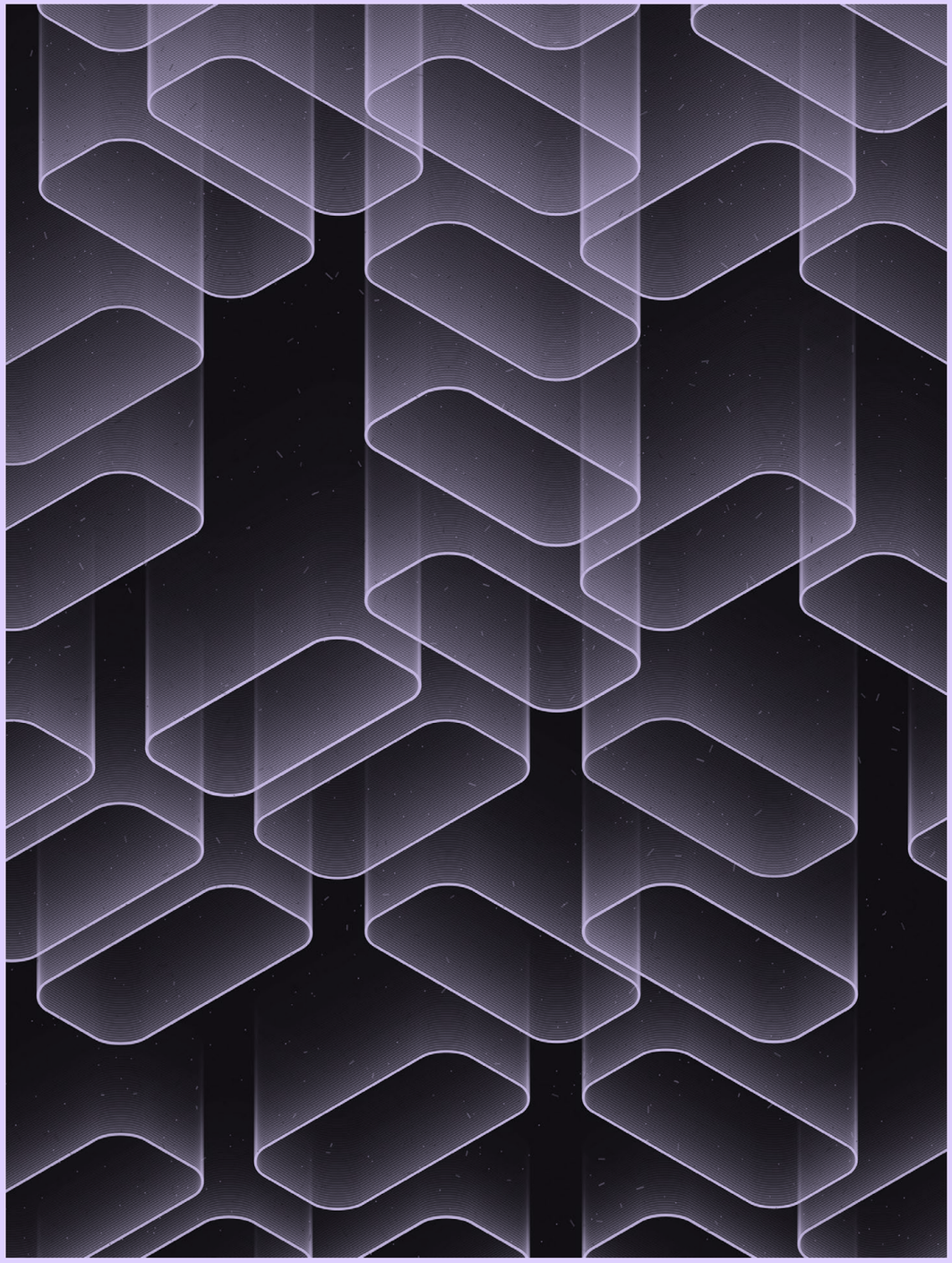


*Common is an experimental videogame exploring community trust in play*

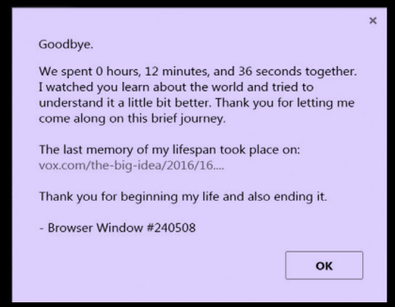



Marie Claire LeBlanc Flanagan

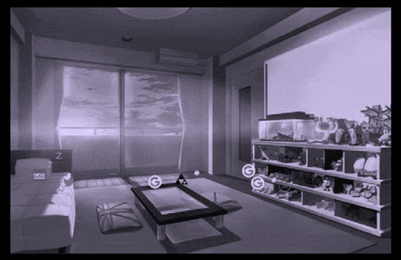




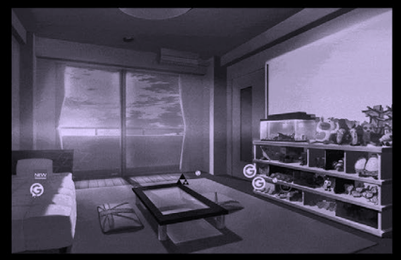

[How might touchscreens express warmth?]

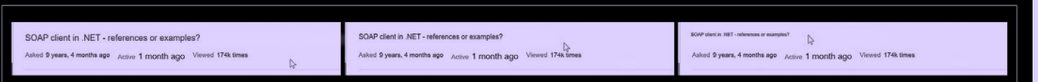



[How might close buttons express connection?]



[How might icons express a bedtime ritual?]





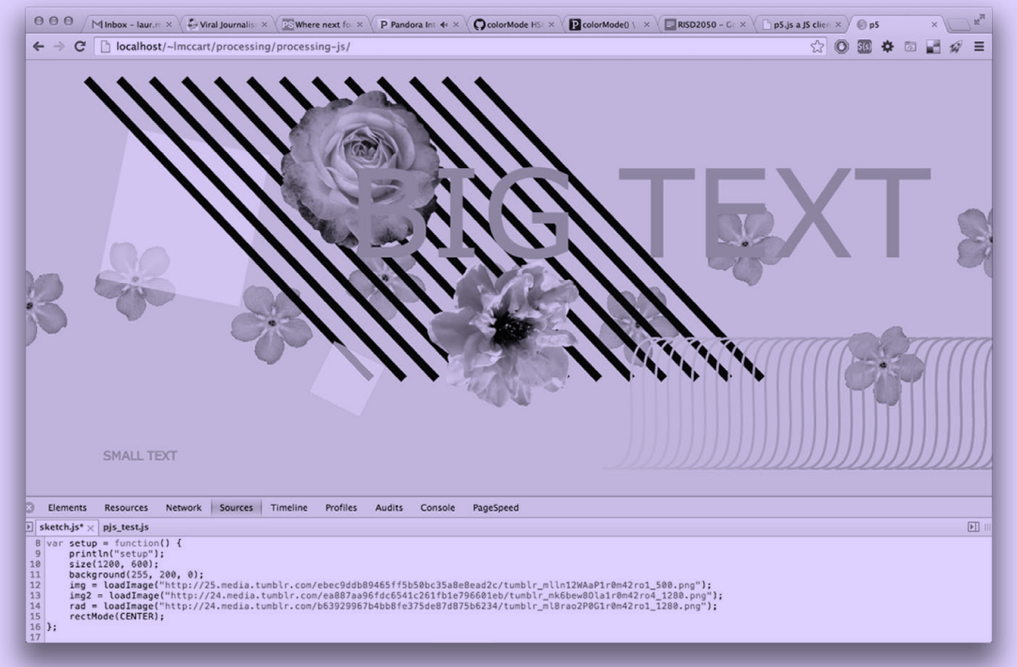

[How might hyperlinks express shyness?]

## "Queering Our Interface"

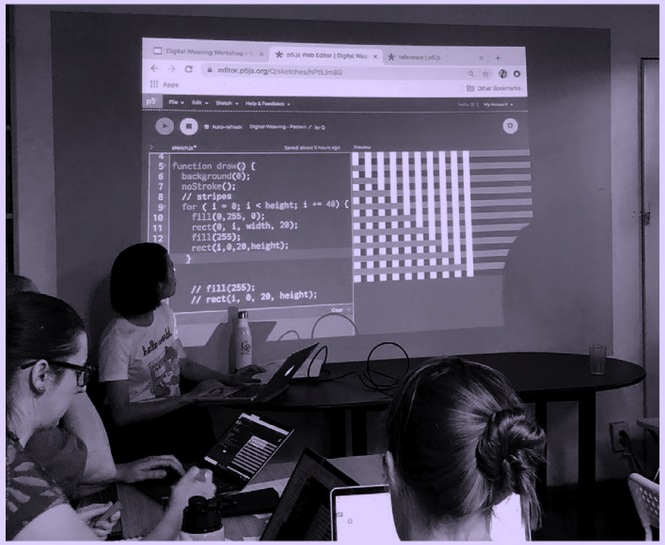
examines the inherent hetero/gendernormative foundation that our computers, and more specifically, our internet's interface is built on. This work serves as both a critique of the current human/computer relationship as a manifestation of hetero/gendernomativity under capitalism, and also an attempt to reimagine what our relationship with the machines we use might look like. Specifically, by reimagining our relationship as one of mutual and non-transactional care, these interfaces are inherently subverting the prevailing markers of technology, namely efficiency, utility, and productivity. How might creating alternative human/computer relationship open up possibilities for relationships dynamics in the physical world?

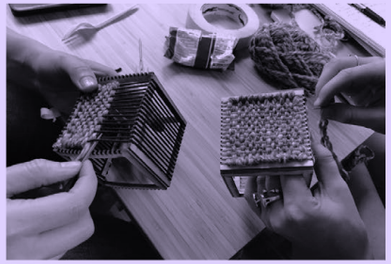
queerinterface.net






SCREEN SHOT 2013-07-04 AT 01.21.09 AM.PNG
P5.JS FIRST WORKING PROTOYPE
LAUREN LEE MCCARTHY

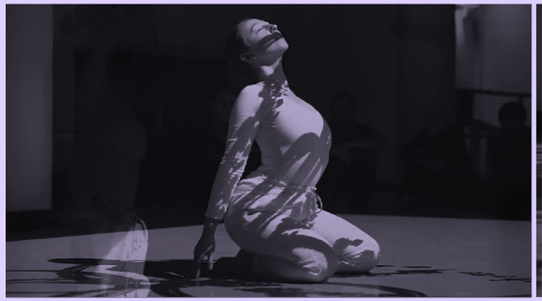


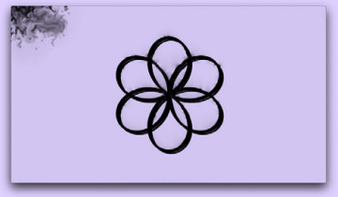



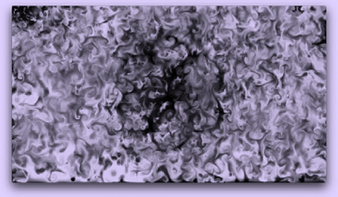
## Digital Weaving, Physical Computing

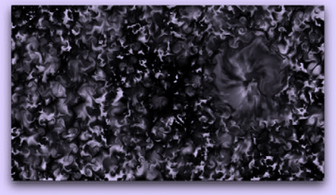
While Artist in Residence at the Feminist Center for Creative Work (formerly Women's Center for Creative Work) in Los Angeles in the fall of 2019, I created a workshop series in creative coding and electronic textiles. During my residency I reactivated the innate connections between weaving and computing by creating and exhibiting weavings, video, and conducting workshops that examined the interrelationships between technology, craft, and women's labor. Qianqian Ye, Evelyn Masso, Berfin Ataman, and I collaborated in teaching a series of three workshops in which students learned to draw and create patterns in p5.js, used data sets to create patterns, made small handweavings, and then used the computational principles they learned to make their weavings interactive with Arduinos.

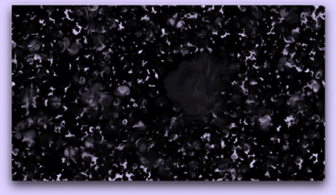
Ahree Lee
www.ahreelee.com







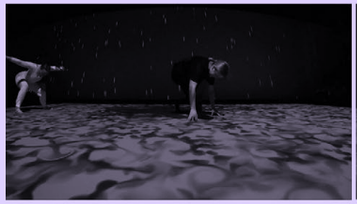


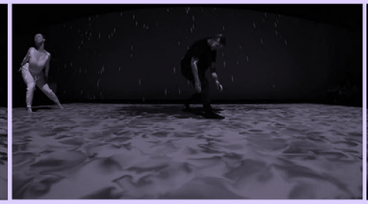


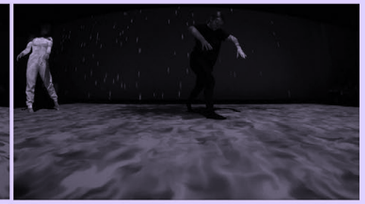
Succumb (2019). Dance performance, New York.
Rain in p5.js, interactive floor in Processing.
Code based on circles and fluid dynamics.

Whenever we built a new art piece, we think of it much as a tool with parameters and stages that can change over time - like a choreographic intervention. Often in response to tracked movements. In Succumb, one dancer draws calm ink patterns, another busier dancer brings in the erratic ink rain that will inevitably destroy the calm patterns of the first dancer. But staying erratic for too long will make you succumb to your own chaos, and eventually leads to a completely black canvas. While staying calm and refinding the patterns that make sense within ourselves - even in a transformed environment in which you feel like you are drowning - makes it possible to resurface. The piece is inspired by the two dancers personal styles of movement. Mentorship by Mimi Yin.

**Code** Louise Lessél Jingyi Wen
**Dancers** Katie Brady Amanda Klepper





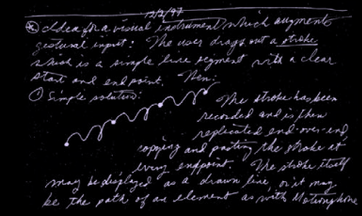




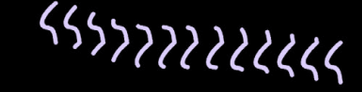
If I had to predict what, from my life's work, might still be around in a hundred years, I think it could be my **Yellowtail** poetic drawing software. If it is, I owe this privilege to Processing, which has included Yellowtail as a pedagogic example since its earliest versions.

Yellowtail is an interactive system for the gestural creation of real-time abstract animation. It repeats a user's marks end-over-end, enabling a user to simultaneously specify both a line's shape as well as its quality of movement. Each line repeats according to its own period, traveling across a screenspace with toroidal boundaries, producing an ever-changing display of lively, worm-like textures. Conceptually, Yellowtail descends from Oskar Fischinger's film studies; expanded drawing tools by John Maeda and Scott Snibbe; and audiovisual instruments by Toshio Iwai.

Conceived in late 1997, Yellowtail became a core component of my master's thesis at MIT, where I gave it to Casey and Ben. It has been performed around the world, projected at Peter Gabriel concerts, and turned into a game (Blek). But most importantly, because of Processing, it has been ported to dozens of toolkits, through which it has given so many people wordless joy and educated them about computational visual form. Thank you, Casey, Ben, and Processing, for this honor. —**Golan Levin**



Above: sketchbook notes (1997) describing Yellowtail. Below: Yellowtail's animation logic.



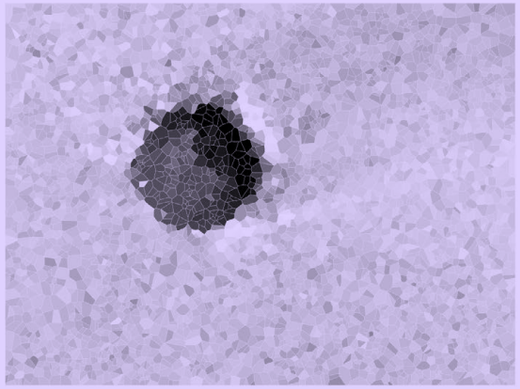
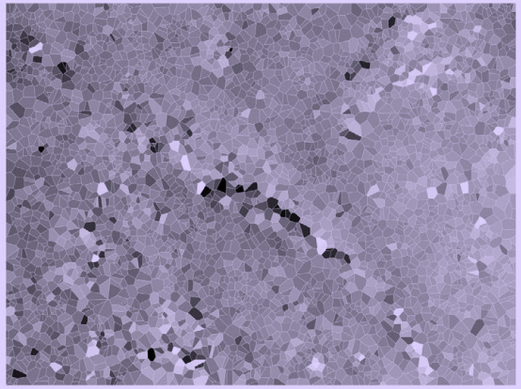
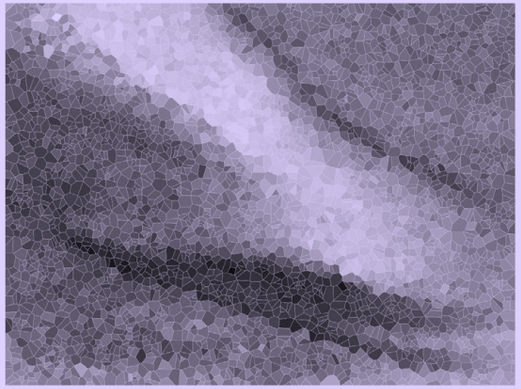
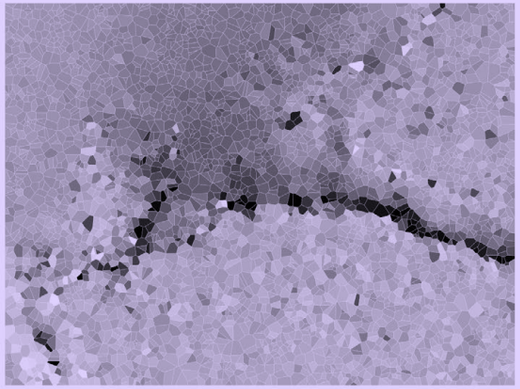
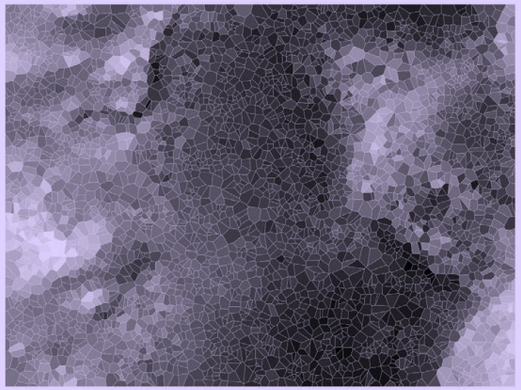
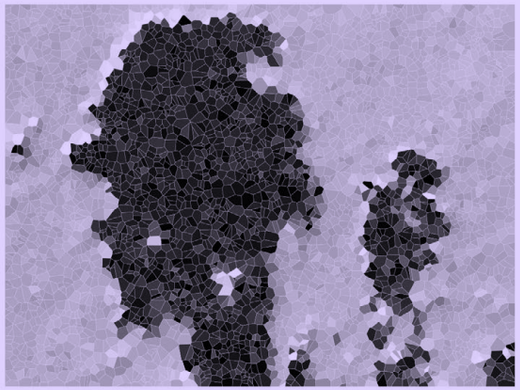



# WithoutTitle

2003 by LIA
(www.liaworks.com)
commissioned by Processing.org
for the online exhibition section.

I learned how to program Java in Processing through this commission - before I used to code Lingo in Director.

I am very grateful that Processing exists. It has been a tremendous help over the last two decades in realising my artistic work.



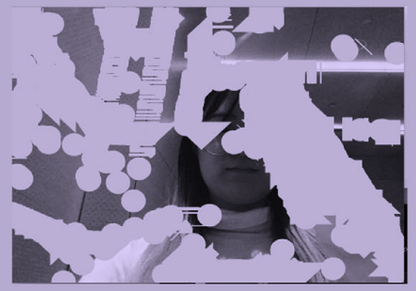
# Machine Vision Self Portraits

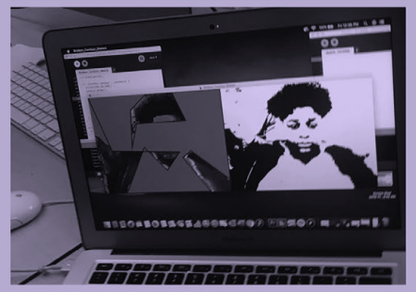
## Workshop at Processing Community Day NYC 2020

February 8, 2020, New School University Center, New York

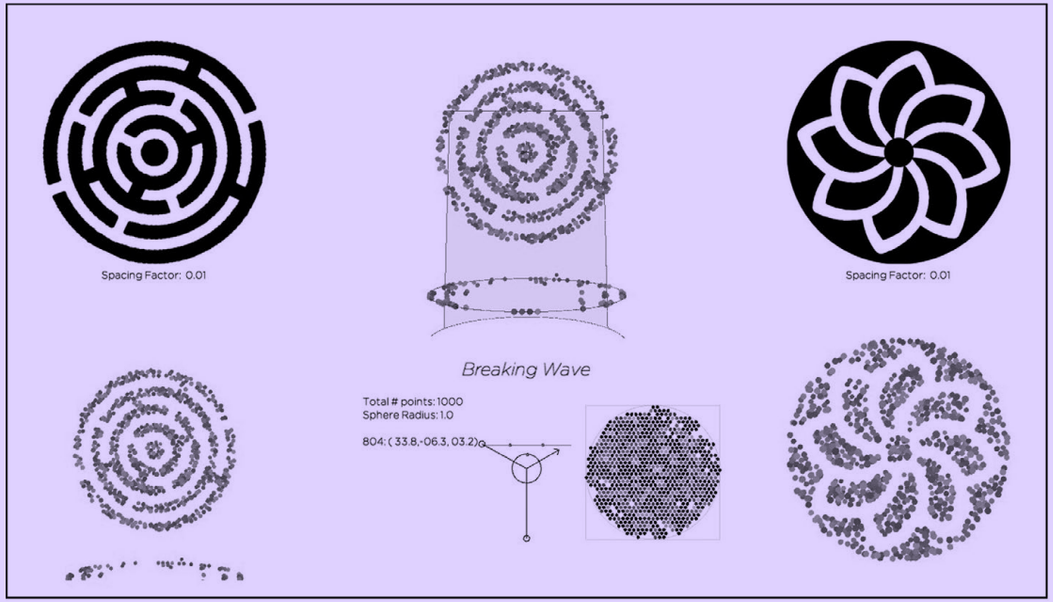
http://processing.nyc/2020/descriptions/index.html#machine_vision

Daniel Lichtman (danielp73@gmail.com) in collaboration with Jose Benitez, Kaitlyn Chiu, Yingna Lu, current New Media Art Students at Baruch College, CUNY



In this workshop we will develop an application in Processing that creates machine vision (MV) self-portraits. Aimed at beginners, we will explore how a computer 'sees' in terms of shapes, colors, lines and mathematical calculations. We will play with a variety of MV algorithms (eg. face and skin detection, moving-object recognition) and fiddle with parameters (eg. which colors constitute 'skin'? what defines a face?) to generate fascinating and unpredictable results. The workshop will cover basic coding in Processing and look at the OpenCV library, which provides easy access to many MV tasks. We will provide code templates to help beginners dive right in, and help individual participants to get their programs running. To point the way towards future conversation about MV's role in society, we will discuss examples such as automated security clearance at Shanghai Airport, smile-to-pay kiosks at KFC and John Deere's 'See and Spray' MV pesticide applicator. We will address how MV relates to capitalism and surveillance, and how it may disempower or empower groups of people--for example how immigrants could be targeted by automated forms of police surveillance or how MV could use photos on social media to determine your insurance rates. We hope to provide participants with a fun and thought provoking introduction to machine vision and creative coding, and to use this as a starting point for thinking critically about MV in society.



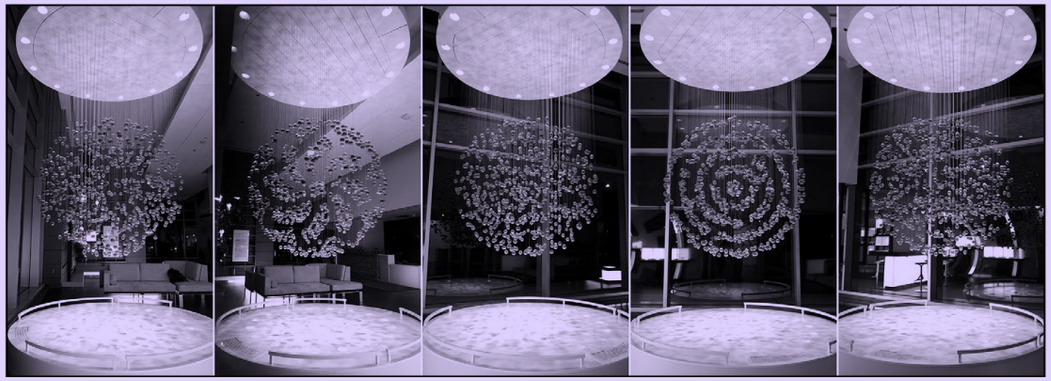

**Breaking Wave** is an anamorphic kinetic sculpture created by Hypersonic and Plebian Design for Biogen-Idec's headquarters in Cambridge, MA in 2014.

804 suspended spheres move in a wave-like formation. When the wave crests and breaks, the balls hover momentarily in a cloud. From almost anywhere in the room, this cloud is purely chaotic, but step into one of two hidden spots, and this apparent chaos shows a hidden pattern. From the first, a labyrinth hints at the search for knowledge, and from the second, a Fibonacci spiral inspired flower reminds us of the natural order and patterns found in nature.

Because this was for a physical sculpture driven by a motor and a fixed mechanical linkage, we used Processing to figure out where these 804 spheres needed to sit in order to form these two hidden images inside the point cloud. We designed a Monte Carlo simulation (shown above): The two goal images were set as the top corner images. Millions of potential spheres were placed on a hexagonal lattice at different heights. An algorithm determined if the sphere placement would contribute positively to the two images or would obfuscate them. If it augmented both images it was added to the collection of final points. The script found a solution in minutes.

For more information see http://bea.st/breaking-wave







English Español 简体中文 한국어 हिन्दी

# p5.js

主页
编辑器
下载
捐献
入门
参考文献
程式库
学习
Teach
范例

书籍
社群
案例展示
论坛
GitHub
Twitter
Instagram
Discord

## 你好!

Search p5js.org

p5.js 是个 JavaScript 创意编程程式库，其专注在让编程更易于使用及更加广泛的包容艺术家、设计师、教育家、初学者以及任何其他人！p5.js 是个免费及开源的软件因为我们相信所有人都应该能自由使用软件及用于学习软件的工具。

p5.js 使用绘图的比喻并有一副完整的绘画功能。除此之外，您也不单限于您的绘图画布。您可以将您整个浏览器页面当作您的绘图，这包括了 HTML5 物件如文字、输入框、视屏、摄像头及音频。

Join the p5.js Discord!

使用 p5 编辑器开始创作!

### 社群

我们声援及支持所有人，不论他们来自不同性别认同、性取向、种族、国籍、语言、神经型、大小、能力、阶级、宗教、文化、次文化、政治观点、年龄、技能等级、职业或背景。我们了解并不是所有人都有多余的时间、金钱或能力以活跃的参与社群活动但我们同时认可及鼓励各种参与方式。我们致力于促进障碍消除及培养能力。我们都是学习者。

p5.js 是 Processing 于今日的网页的诠释。我们的活动及运作由 Processing Foundation 所支持。

了解更多关于我们的社群。

### 入门

使用 p5.js Editor 制作您的第一个绘图。以了解更多关于如何使用 p5.js 绘图及其各种功能请参考入门页面及参考文献。

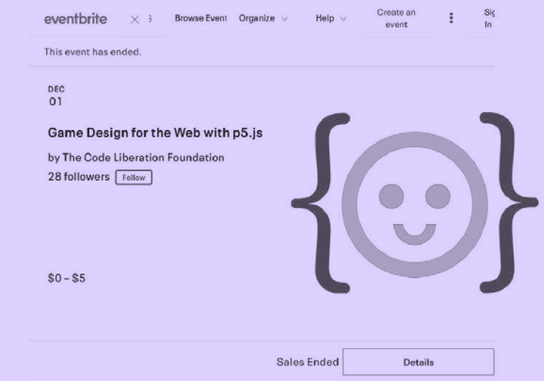
### 参与

我们有各种方式让您贡献给 p5.js：

- 分享您制作的项目!
- 使用 p5.js 教学。
- 举办个聚会。
- 贡献代码。

p5.js 目前由 Qianqian Ye 和 evelyn masso 的原 Lauren Lee McCarthy 并在 Processing Foundation 和 NYU ITP. 的支持下由其合作社群所开发。形象及平面设计 Jerel Johnson. © Info.






to generate playful environments. There are many sophisticated game engines in the market, however it can be cumbersome to produce working builds to be played by a mass audience. So why not utilize a medium that already engages your audience everyday? These workshops aim to give you the skills to make games that anyone with an internet browser can access!

p5.js is a new open source javascript library developed by the Processing foundation to make coding more accesssible for artists, designers and educators. We will be using p5.play - a p5.js library created by Paolo Pedercini to work with sprites, simple animation, collisions and basic user interaction to develop a game. We will also show you a quick way to get your project onto the web.

**Instructors**

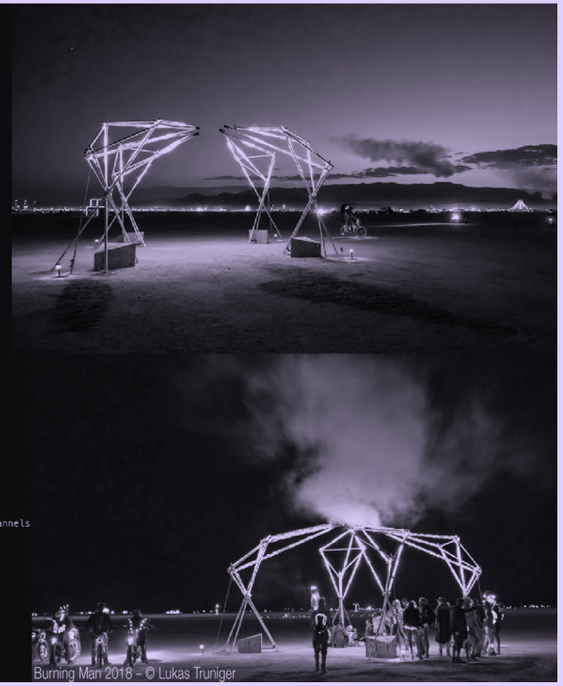
Adelle Lin creates experiences that integrate the digital and physical in playful ways. By day she produces interactives for the New-York Historical Society Museum and by night is a playful unicorn at NYC Resistor. Working with Open Frameworks, Unity, VR and physical computing elements, she has exhibited games at Burning Man, A Maze. Festival, Maker Faire New York, Come Out and Play Festival, and Different Games Conference. Having lived in Singapore, Malaysia and Australia, she sees the world through her multi-colored glasses.

Rachél Bezalais is a front-end web developer and indie game designer. She is a programmer focused on intertwining interactive experiences with responsive and efficient design. She aims to design games with the capacity to tell stories through gameplay that continue to push boundaries of how we perceive the capabilities of the medium. Aside from building all the things, Rachél spends a portion of her time teaching game programming with the Code Liberation Foundation.

Tags

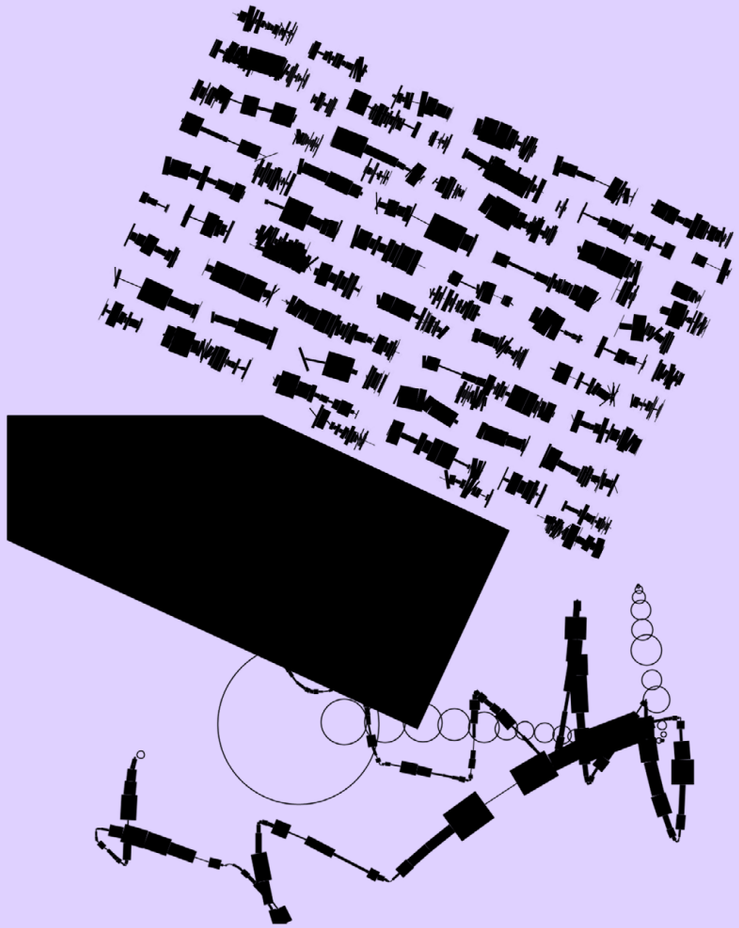
United States Events | New York Events | Things to do in New York, NY | New York Classes | New York Other Classes

```
void draw()
{
  // create color
  int c = color(frameCount % 360, 80, 100);

  background(0);
  stroke(0);

  if (readAnemometerSerial == true) {
    readAnemometer();
  }

  //change direction
  //if (ellapseTimeMs[0]> durationMs) direction = !direction;

  drawImageToScreen();

  loadPixels();

  drawPatternToScreen();

  // fog pattern and draw
  poof = poofEvent.updatePoofEvent();
  flood = poofEvent.updateFloodEvent();
  updateFogPixels(poof);
  updateFloodPixels(flood);

  // read pattern from screen draw
  if (readFromScreen == true) {
    updatePixelBufferFromScreen();
  }

  LedArtnetclass.updateArtnet(artnet, dmxData, pixelBuffer, numPixelUniverse, numLedChannels
  LedArtnetclass.sendArtnet(dmxData, numPixelUniverse);

  updateEllapseTime();

  drawPixelBuffer();

  println(frameRate);
}  // end draw()
```

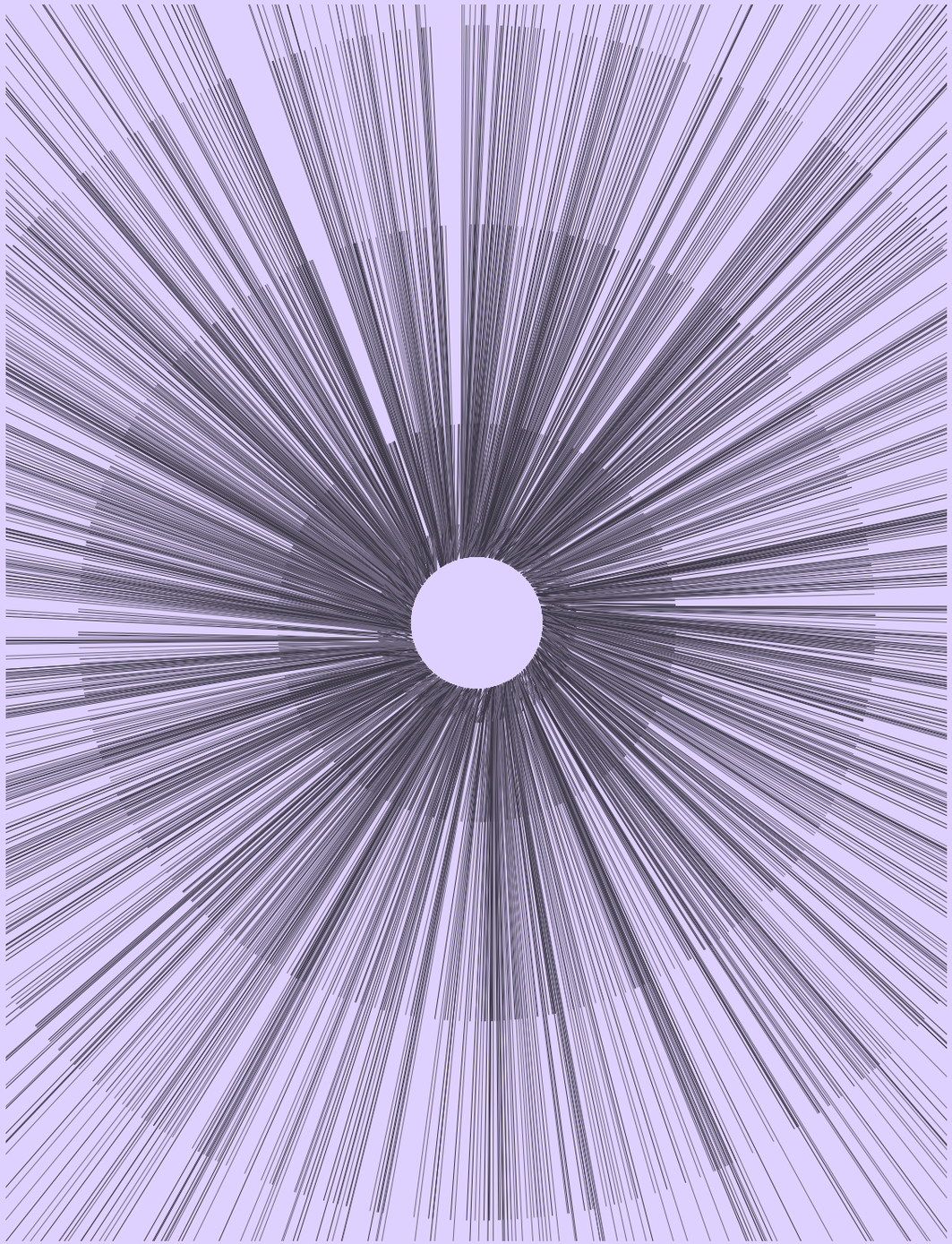

Burning Man 2018 – © Lukas Truniger

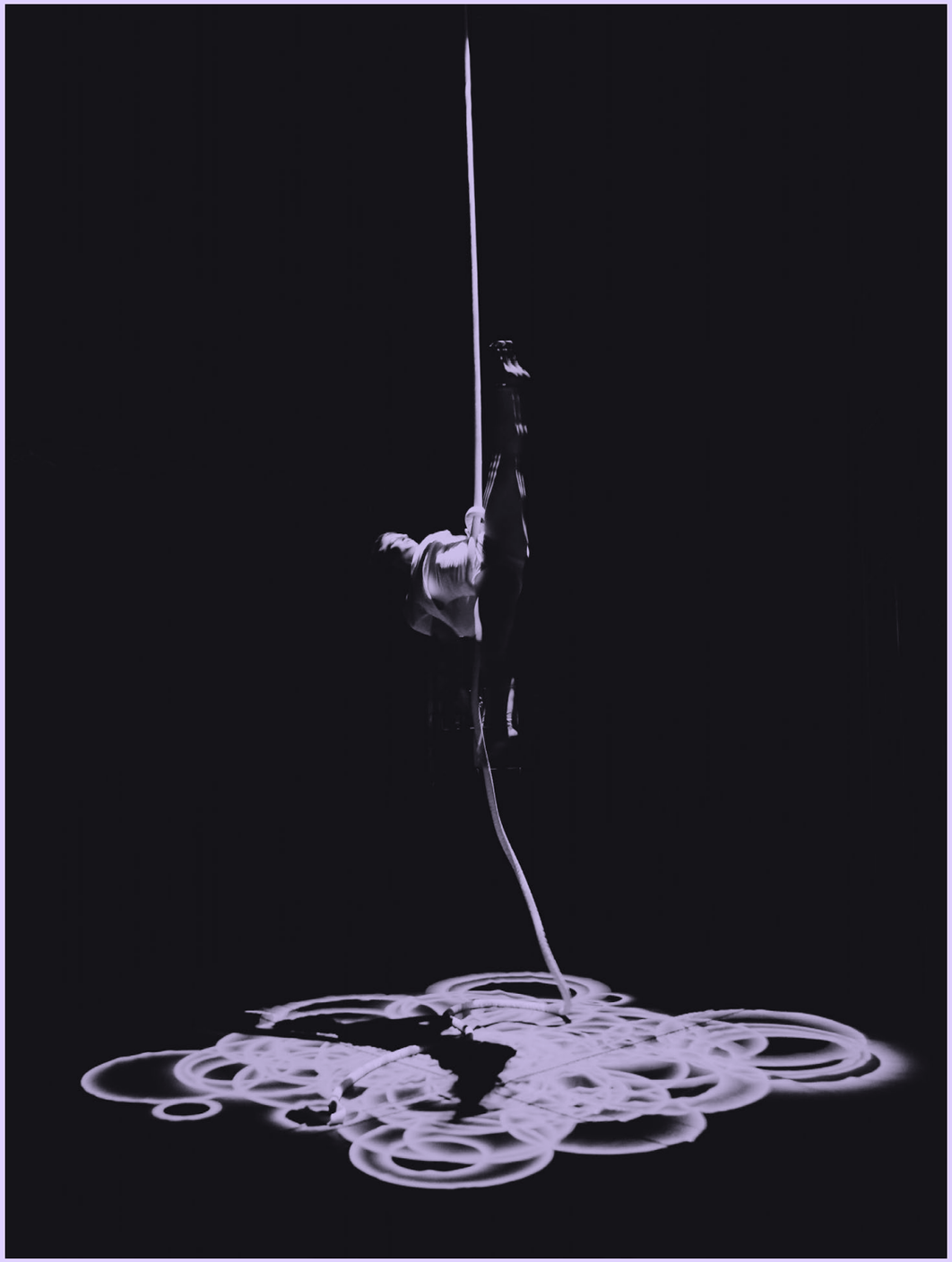
Dear Processing , It has taken ages for me to move from just an avid fan to a practicioner, to a teacher and now a proselytizer of your ways. As a tool and as a community you have changed my life, my aspirations and my ways of thinking. Life and all of art-making now seem full of variables, and loops, and conditionals, and arrays. I still struggle to think in terms of objects but I bet it will happen one day. Lately I keep coming back to waves and mapping. I learned to iterate, to pay attention and really stick with practicing because of you. My indie music heroes have been joined by coders thanks to you. I'm so glad that you are now inspiring and producing zines and catalogs. Long may your freak flag fly.

Justin Lincoln
fyprocessing.tumblr.com
https://www.instagram.com/thebuildingisacamera











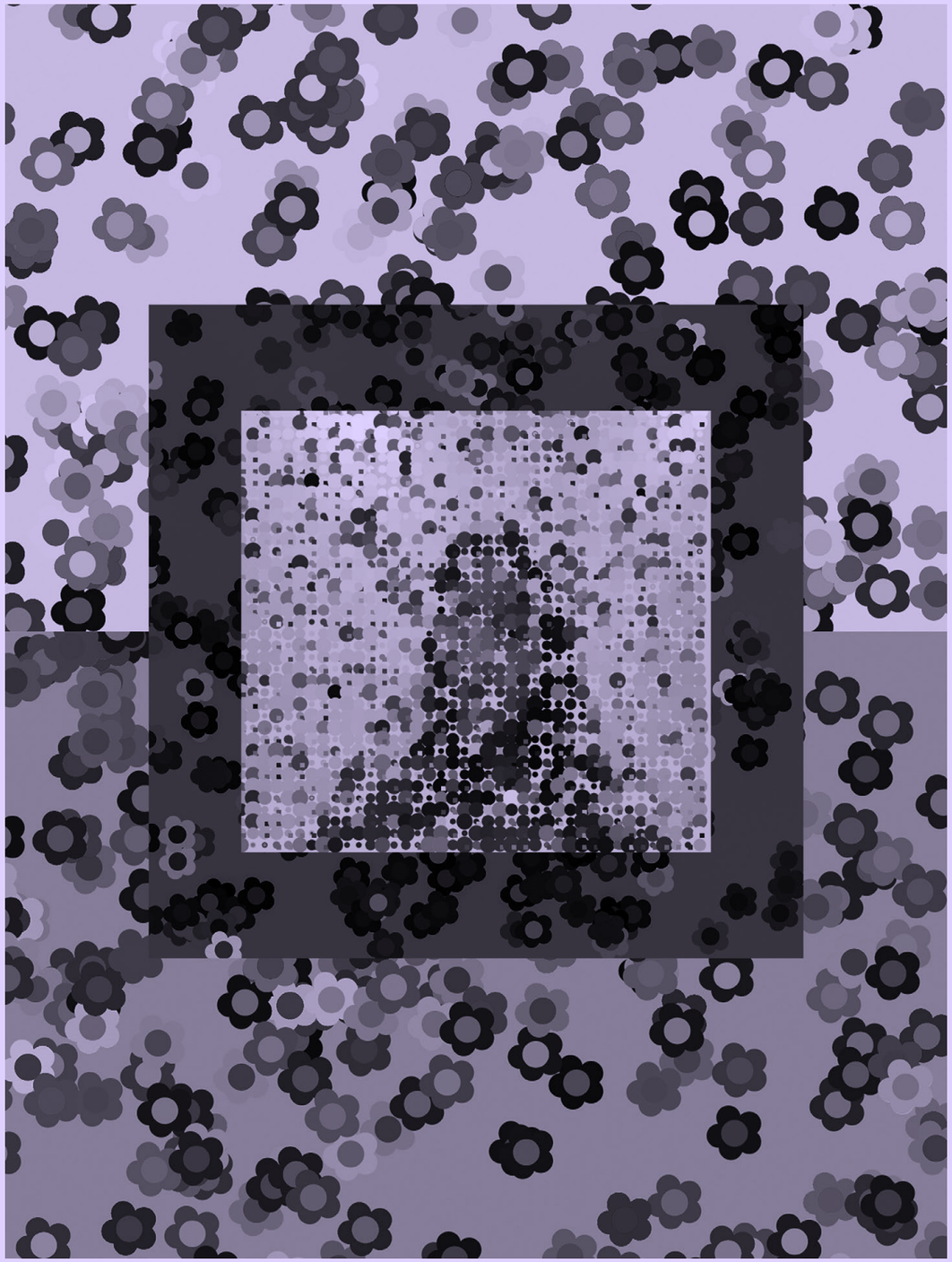
# p5.js 2020 Showcase

showcase.p5js.org

I first stumbled upon p5 my junior year of high school. It opened my eyes to the idea I could blend my passion with art and creating together with computer science.

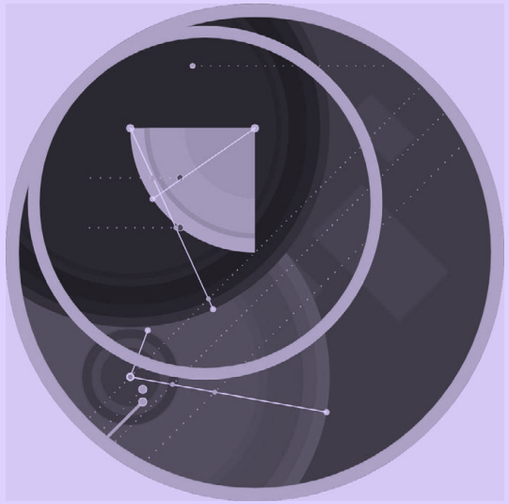
My fears of imposter syndrome melted away after exploring p5 some more, and I was beyond excited to give back to the community with my Google Summer of Code project.

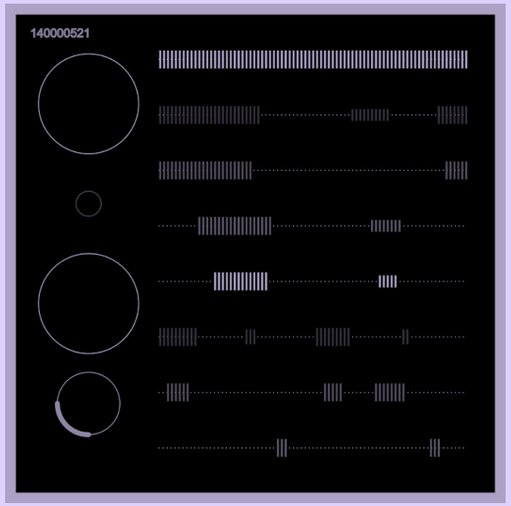
Through my GSoC project I realized how welcoming the community was both through my mentors and many, many others. P5 truly symbolizes that creative coding is for everyone! It was amazing to see how many people were affected by my project from all areas of the world. I can't wait to see how p5 and the Processing community grows in coming years!





A JOURNEY
WITH PROCESSING
**LOACKME**

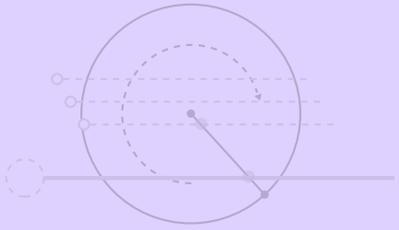


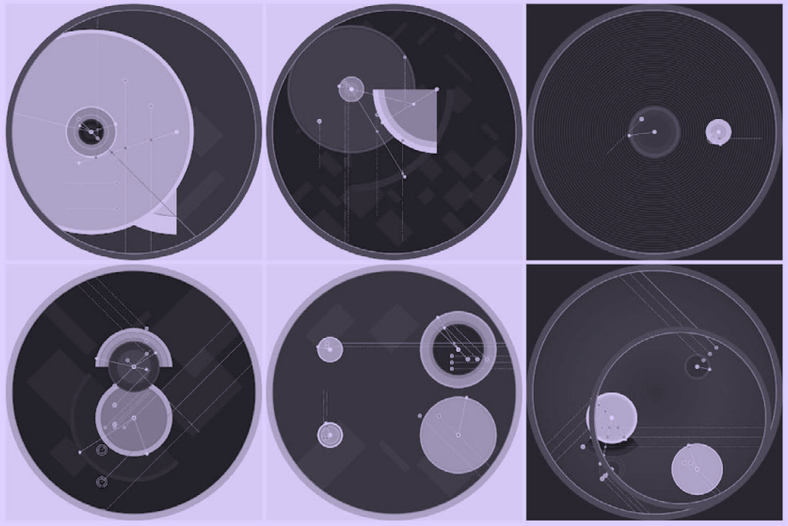



#521

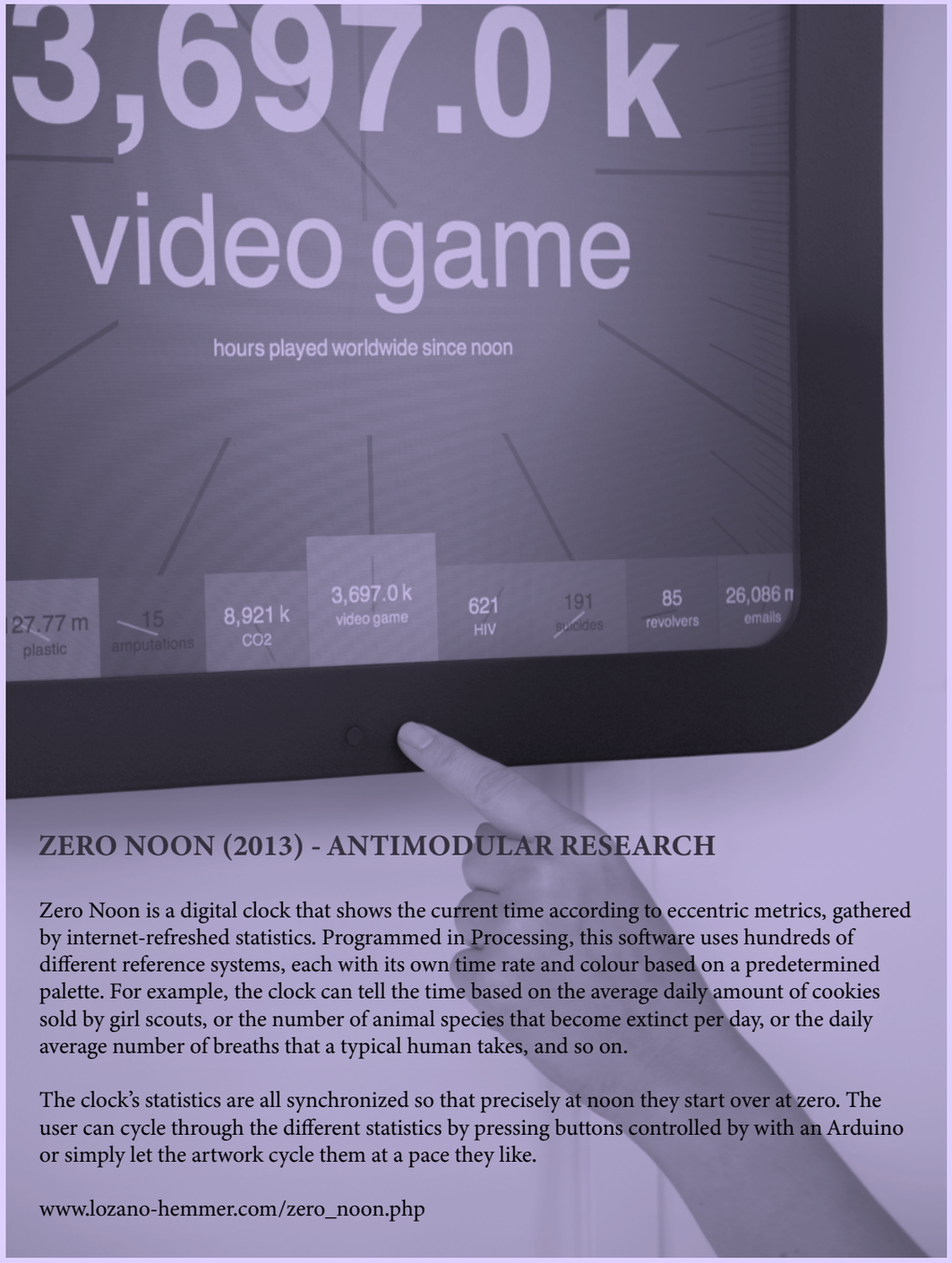
# Good Vibrations & B-side

Aluan Wang https://artblocks.io/project/140







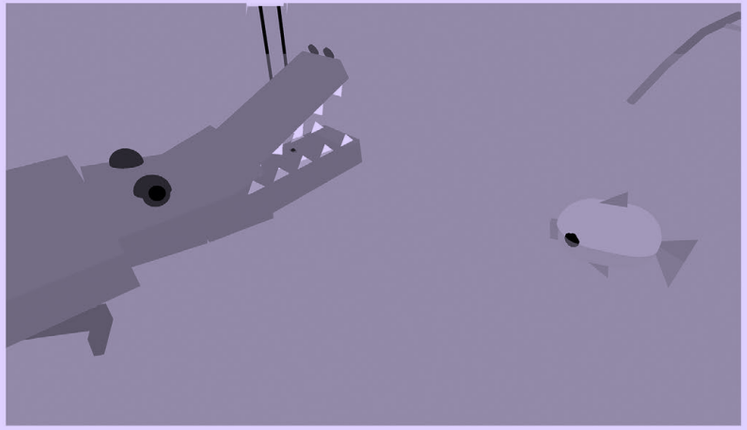



## ZERO NOON (2013) - ANTIMODULAR RESEARCH

Zero Noon is a digital clock that shows the current time according to eccentric metrics, gathered by internet-refreshed statistics. Programmed in Processing, this software uses hundreds of different reference systems, each with its own time rate and colour based on a predetermined palette. For example, the clock can tell the time based on the average daily amount of cookies sold by girl scouts, or the number of animal species that become extinct per day, or the daily average number of breaths that a typical human takes, and so on.

The clock's statistics are all synchronized so that precisely at noon they start over at zero. The user can cycle through the different statistics by pressing buttons controlled by with an Arduino or simply let the artwork cycle them at a pace they like.

www.lozano-hemmer.com/zero_noon.php

Eat Me, Kristin Lucas. Built with p5.js.

Eat Me was commissioned as part of the ENTER program by the Onassis Foundation.



# Homage to B. Franklin

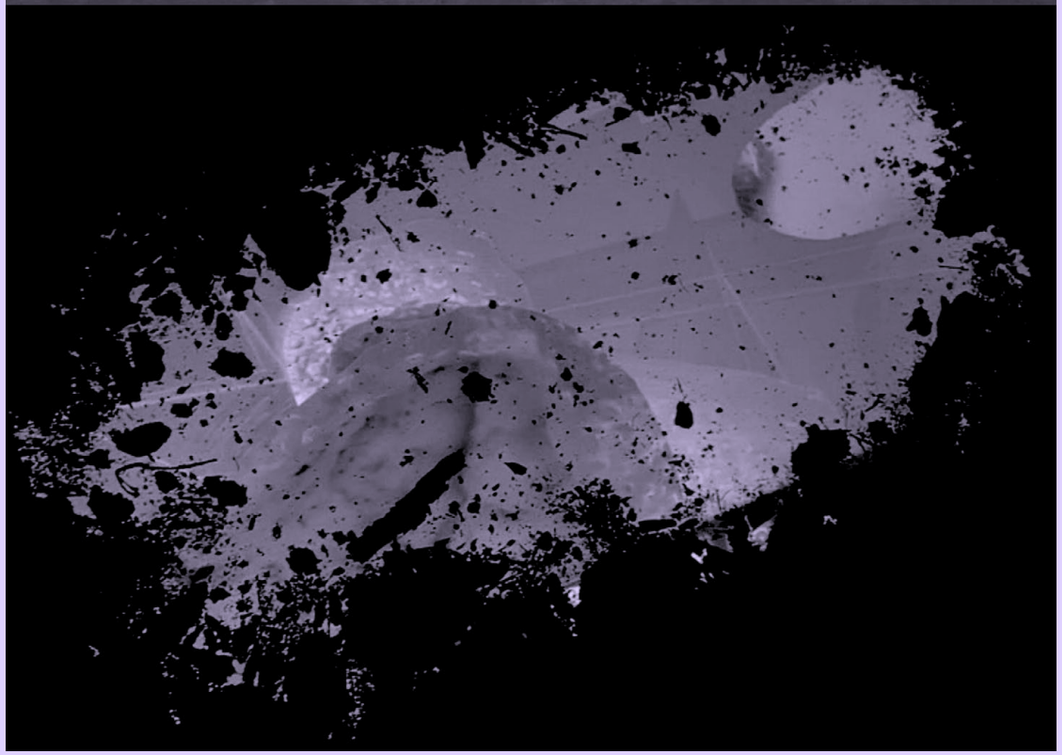
2013, Madrid/ Spain
LUMEN Collective (Nacho Cossío + Fernanda Ramos)
Made with Processing

In 1761, fascinated by glass harps, Benjamin Franklin invented the glass harmonica, an instrument which despite having attracted the attention of big names like Mozart, Strauss, Beethoven, Donizetti and Goethe, was misunderstood and doomed to oblivion.

**Homage to B. Franklin** is an interactive sound installation. Like Franklin, we also got inspired by the glass harps, so popular in the eighteenth century and still enjoyed nowadays. With this reference in mind, we aim to propose a dialogue between the past, exploring sonority from elements of everyday life -like simple crystal glasses filled with water- and the present, using electronic synthesizers and tangible digital interfaces to generate music. In both cases, there is something magical about being able to generate music without necessarily being an instrumentalist, without making use of classical or conventional instruments. In both, the resulting sound is different from what these instruments can generate, but still music.

New sound patterns can be defined by moving the bare hands over the glasses. Stimulating the senses, new creative possibilities emerge through the individual experience of each viewer. In that game generated by the representation of something so old, we propose a reinterpretation and a homage to Franklin and his harmonica.

//This project was first exhibited at *FILE/ SP* 2013 and it is included in *Highlike Book*, a selection of 2013 most liked artworks, published in Brazil by SESI-SP Editorial (2014)






Un encuentro donde artistas visuales, programadores, músicos y poetas presentan procesos y trabajos creativos basados en código.

**Charlas**

Marcos Wasem (Montevideo)
Rosana García Collares (Montevideo)
Cristian Reynaga (Buenos Aires)
Laura Bardier (Montevideo / Nueva York)
Daniel Argente (Montevideo)
Agustín Ramos Anzorena (Buenos Aires)
y Mario Alberto Guzmán Cerdio (México)
Luisa Pereira (Montevideo / Nueva York)
Álvaro Cassinelli (Montevideo / Hong Kong)

**Microconciertos**

Juanita Fernández. Al Térico.
Marcos Giménez. Instrumentos Prototípicos
Pati Horowitz. Phoro Live
Marcos Umpiérrez y Daniel Argente. Esto no es Magritte
Brian Mackern. Soundcode Sketches 1998-2000
Virginia Arigón. Transpiro
Pol Villasuso & Mathias Chumino. Attractor - Repeller

Montevideo Cultura · Cabildo de Montevideo · PCD Processing Community Day



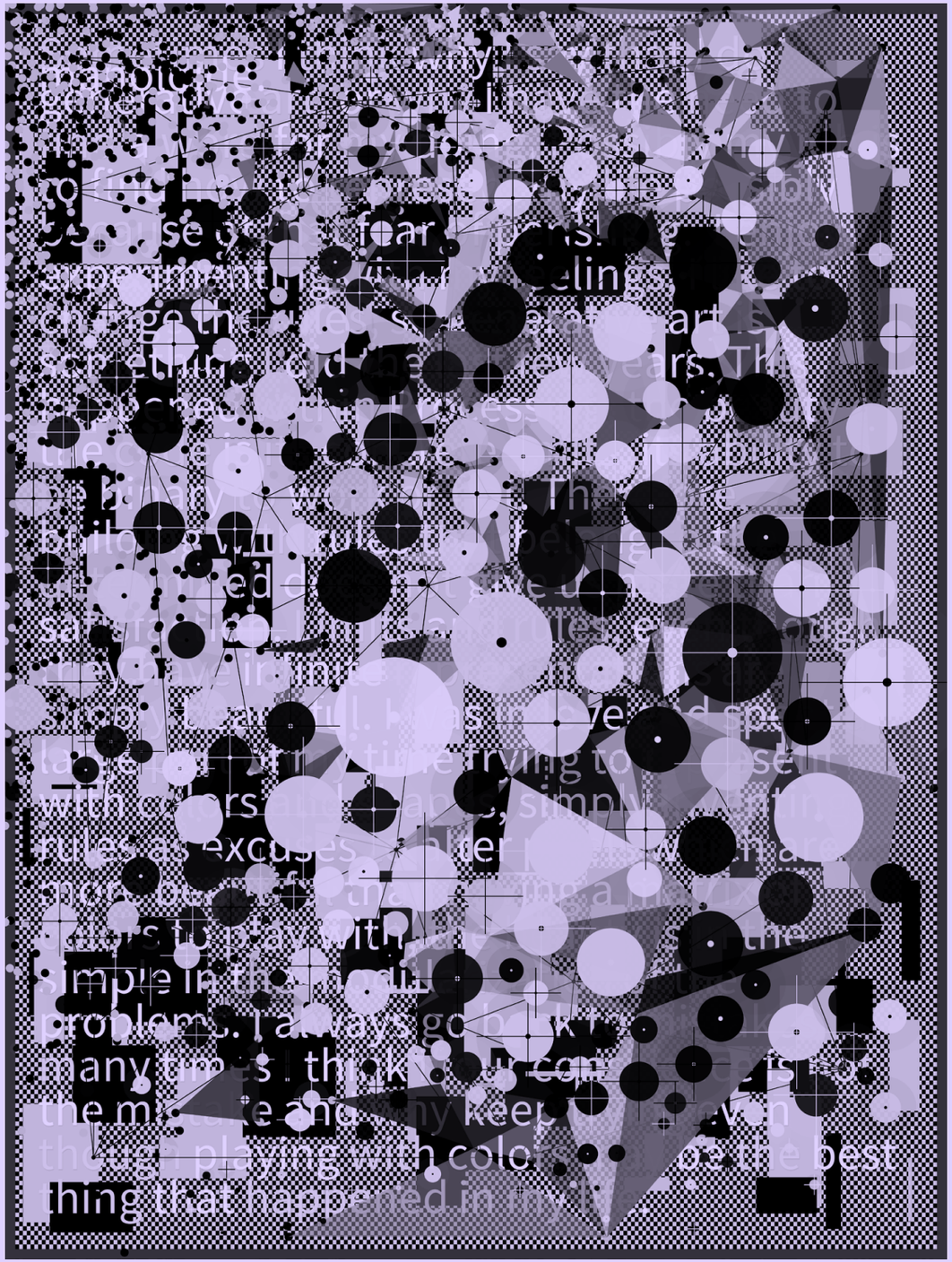
I discovered Processing by chance in 2009 during an optional course in design school. This encounter opened me to a world of infinite possibilities and quickly changed my life like no other software. I owe P5 countless hours of learning and experimentation, my passion for generative art, my career as a creative coder, teacher and artist, as well as many friendships within an incredible community of passionate individuals helping each other.

Long live P5!

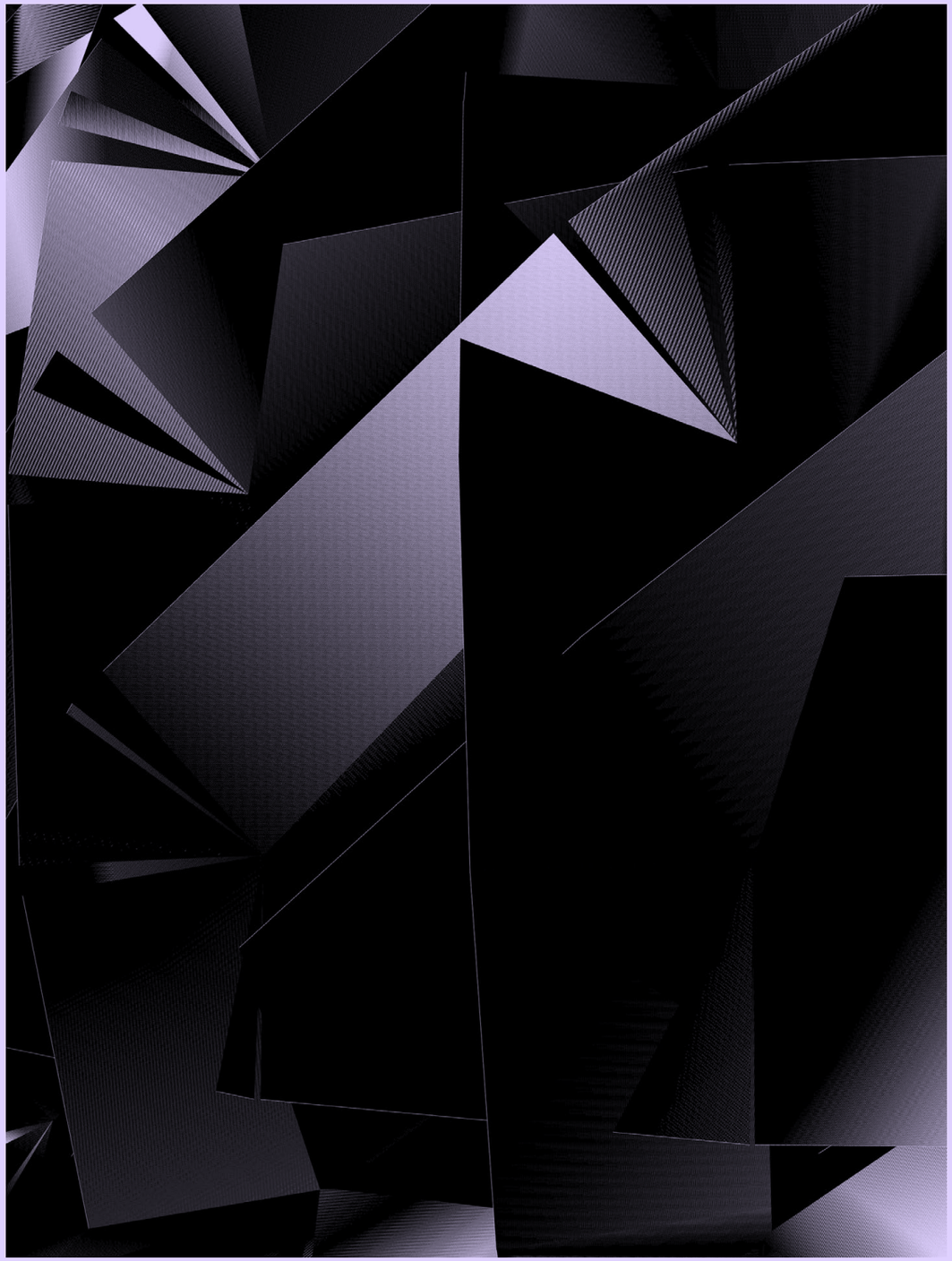
Lionel Radisson | @Makio135


/**
* HELLOOooo00o0o0000ooo
* Code is written by me, Maya... comments link to references where necessary
* Thanks for visiting <3
*/
mayaman
Sketches
Collections
Assets
New sketch
Search sketches...
Sketch
Date Created
Date Updated
[SHAPING YOURSELF ON SCREEN] PoseNet x p5.js
Nov 18, 2020, 3:26:16 PM
Nov 18, 2020, 4:16:44 PM
Maya's M00D RING
Sep 24, 2020, 7:20:31 AM
Sep 24, 2020, 8:04:00 AM
[MAYA] Simulate: SoftBody
Sep 23, 2020, 6:08:35 PM
Sep 23, 2020, 6:09:24 PM
Paint me in pixels so I can dance forever II
from Maya Man
00:53
vimeo
MANIFEST YOUR DREAMS ON SCREEN!
AN INTRODUCTION TO CODING WITH P5.JS
#SupportP5 Artist: Maya Man
Processing Foundation
Follow
Dec 30, 2019 · 5 min read
Maya Man
can I go where you go?
Custom software (color, silent), computer, screen or projector
Dimensions variable, horizontal or vertical
Generative, interactive
JavaScript (p5.js), HTML, CSS
Web browser, Linux, Windows, Mac
Edition of 76, 1AP
View Artwork
Collect this Edition
Maya Man's unique contribution to Social Codes offers a new way to think about software and performance. Her deep explorations in art, dance, and code come together in "can I go where you go?" As a video clip of Maya dancing plays, her code analyzes the video frames one by one to generate a silhouette. This contour becomes an ever-changing shape the viewer can take charge of by drawing with it, or they can simply watch as the software performs itself.

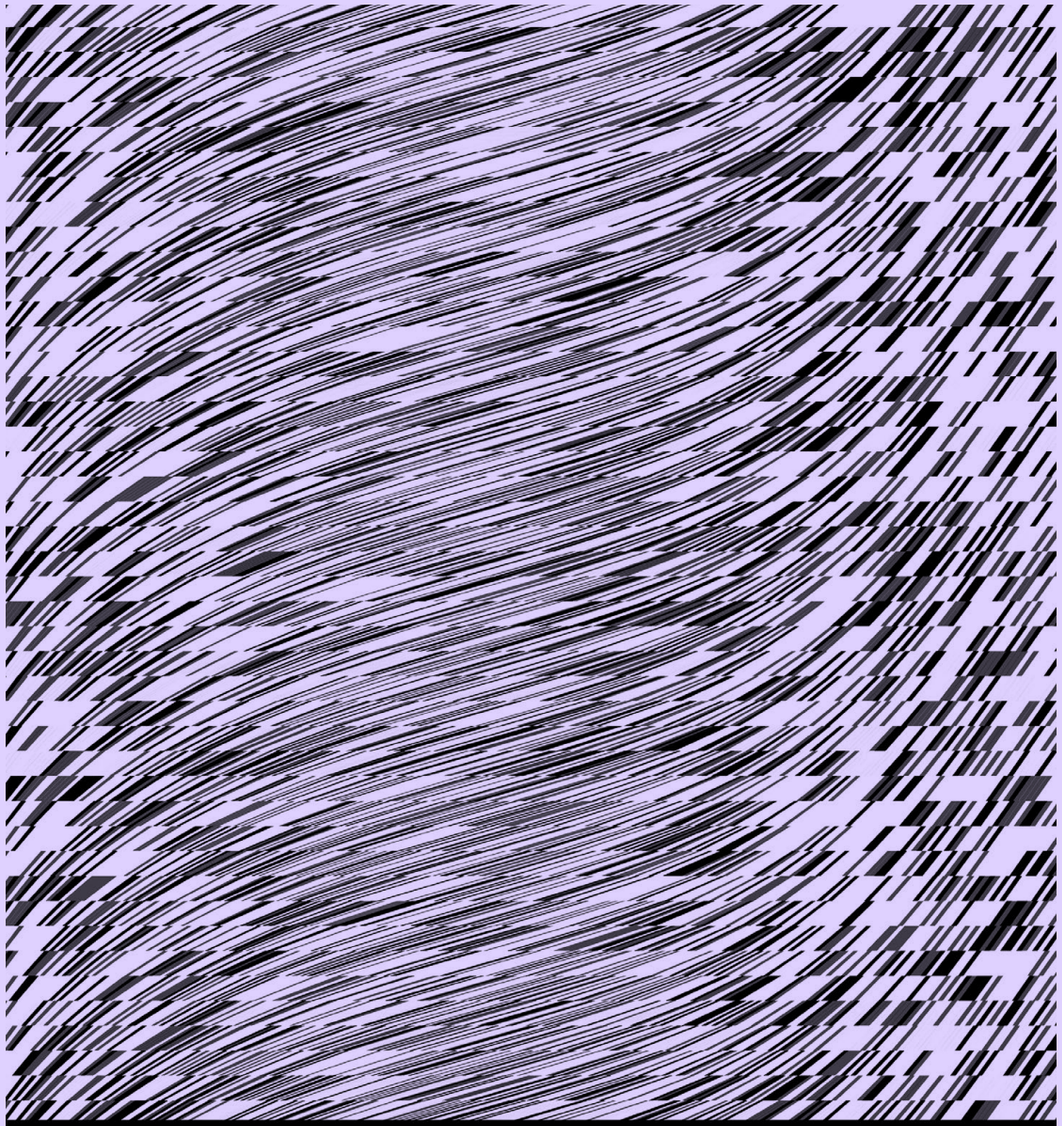

UNIVERSO
DE
CONTATO



MANOLOIDE.COM CON 329



VINICIUS MANRIQUE CON 330

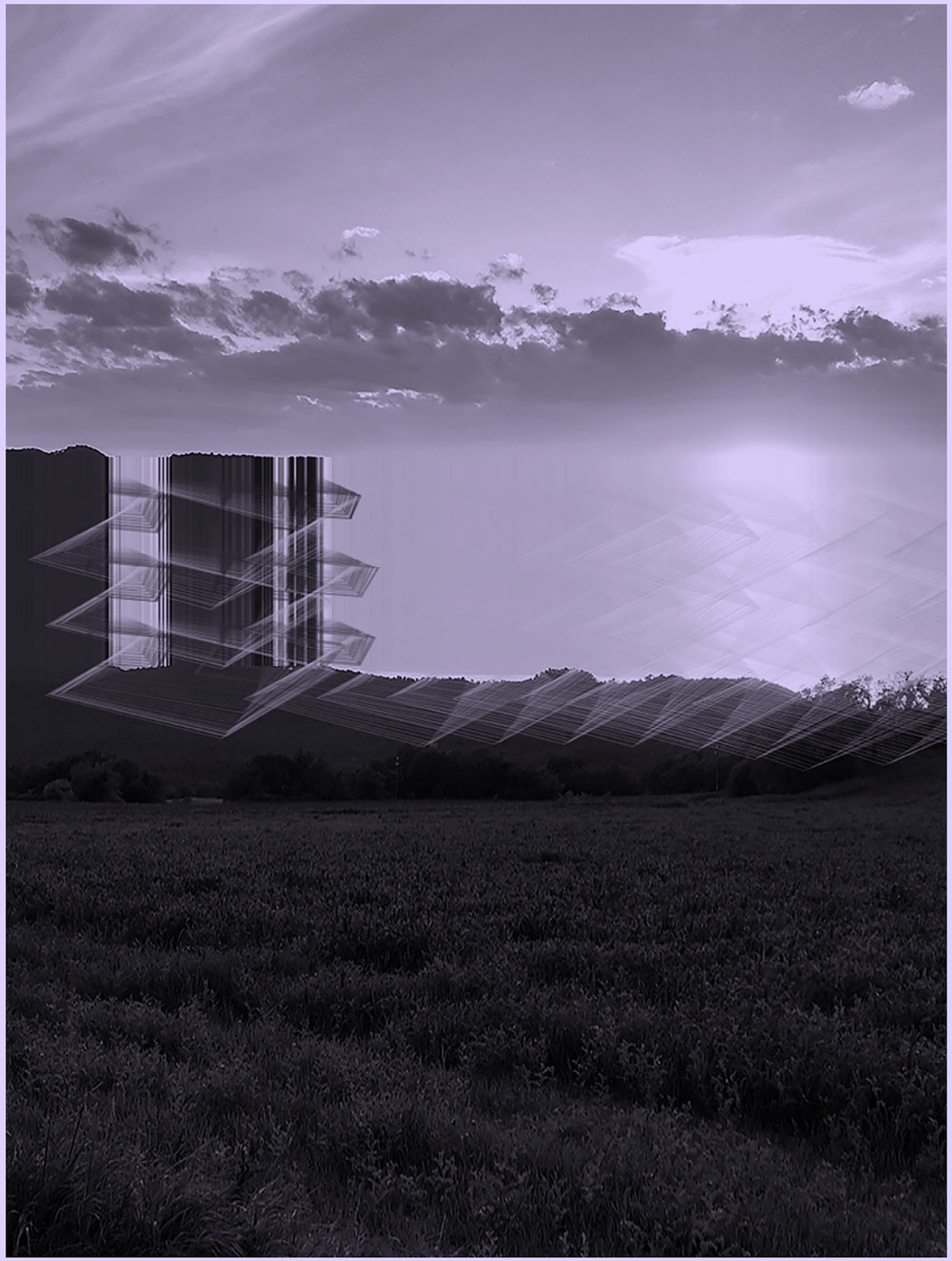


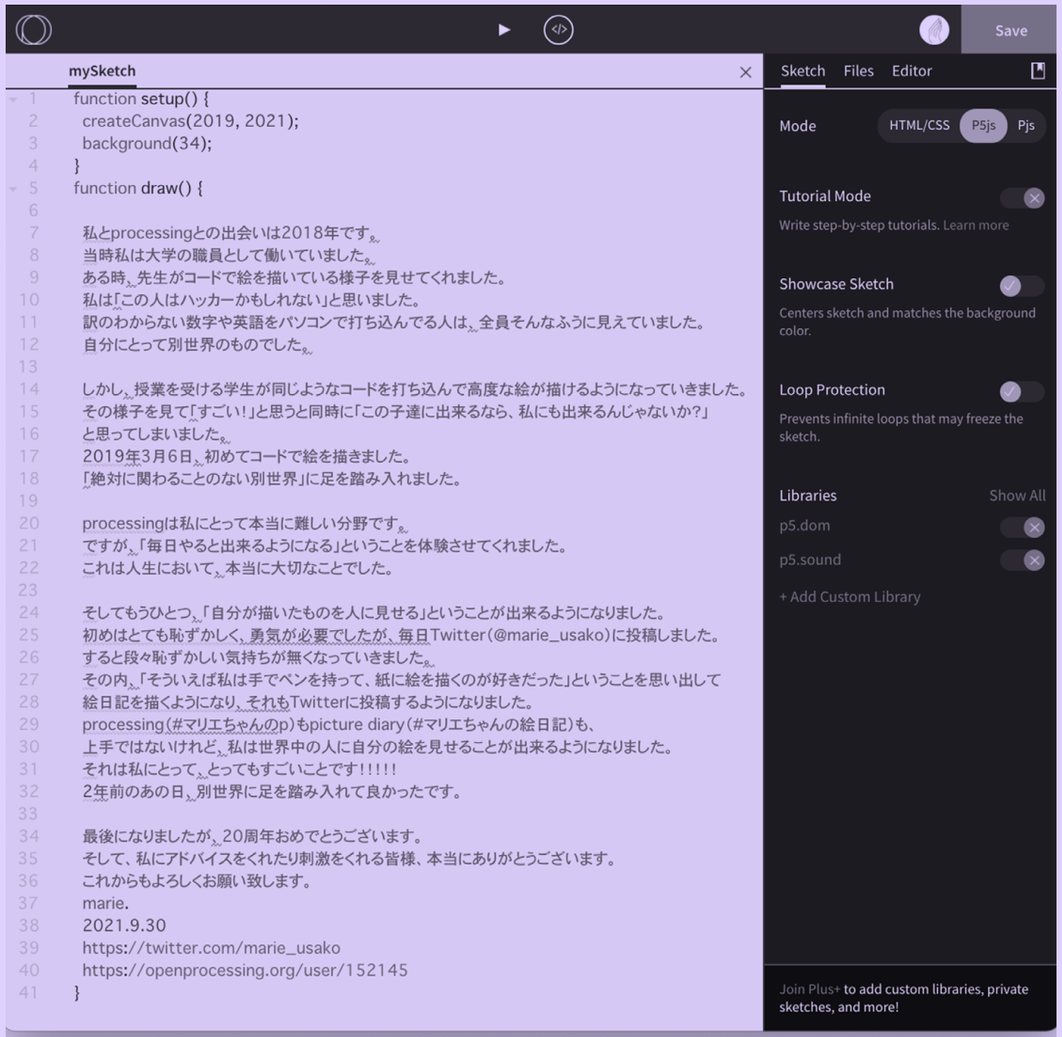
# Happy 20th Birthday, Processing!

Thank you for building such a great software.
Enabling us to learn, share and have fun within a great community.

William Mapan (@williamapan)

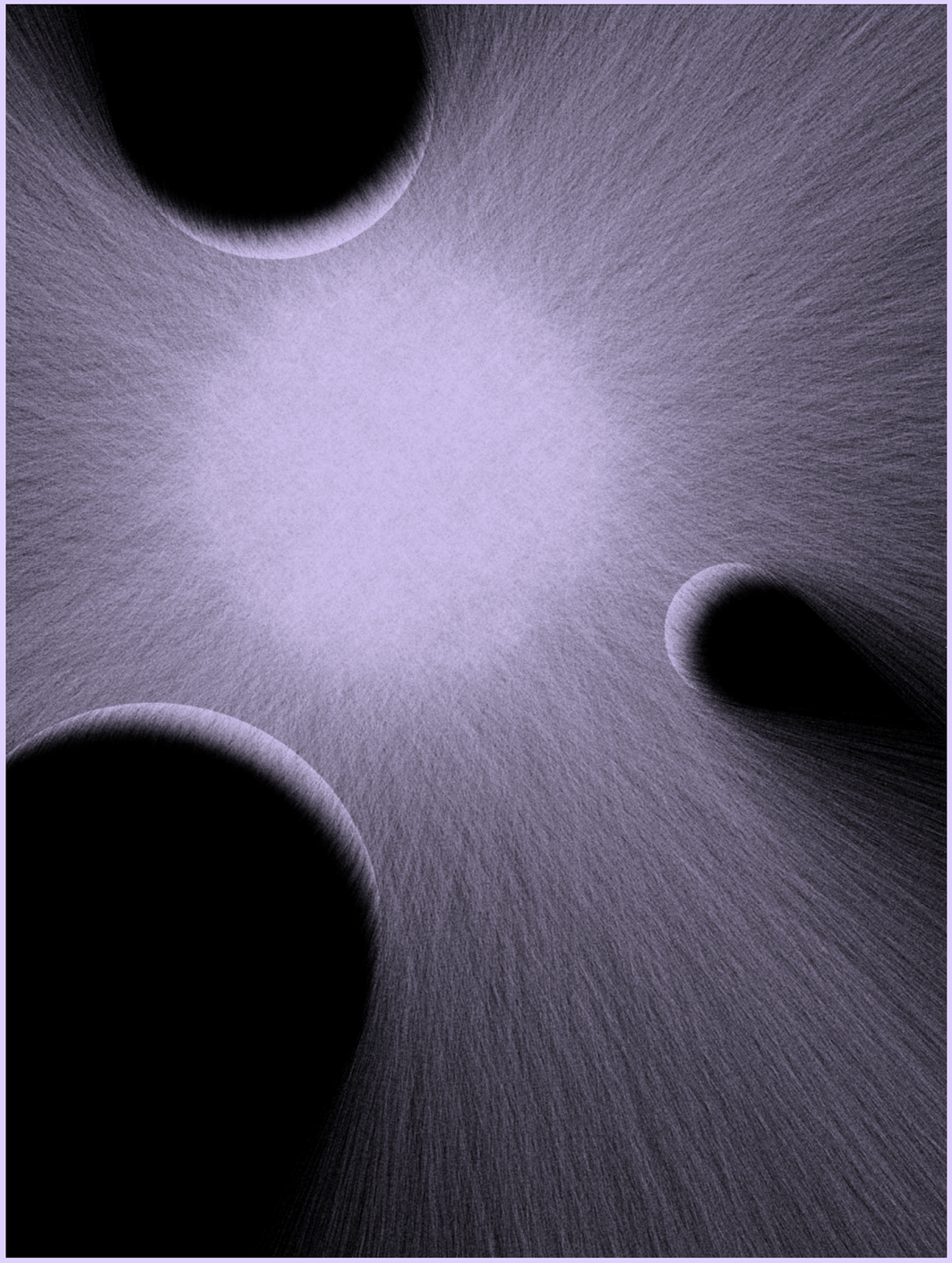


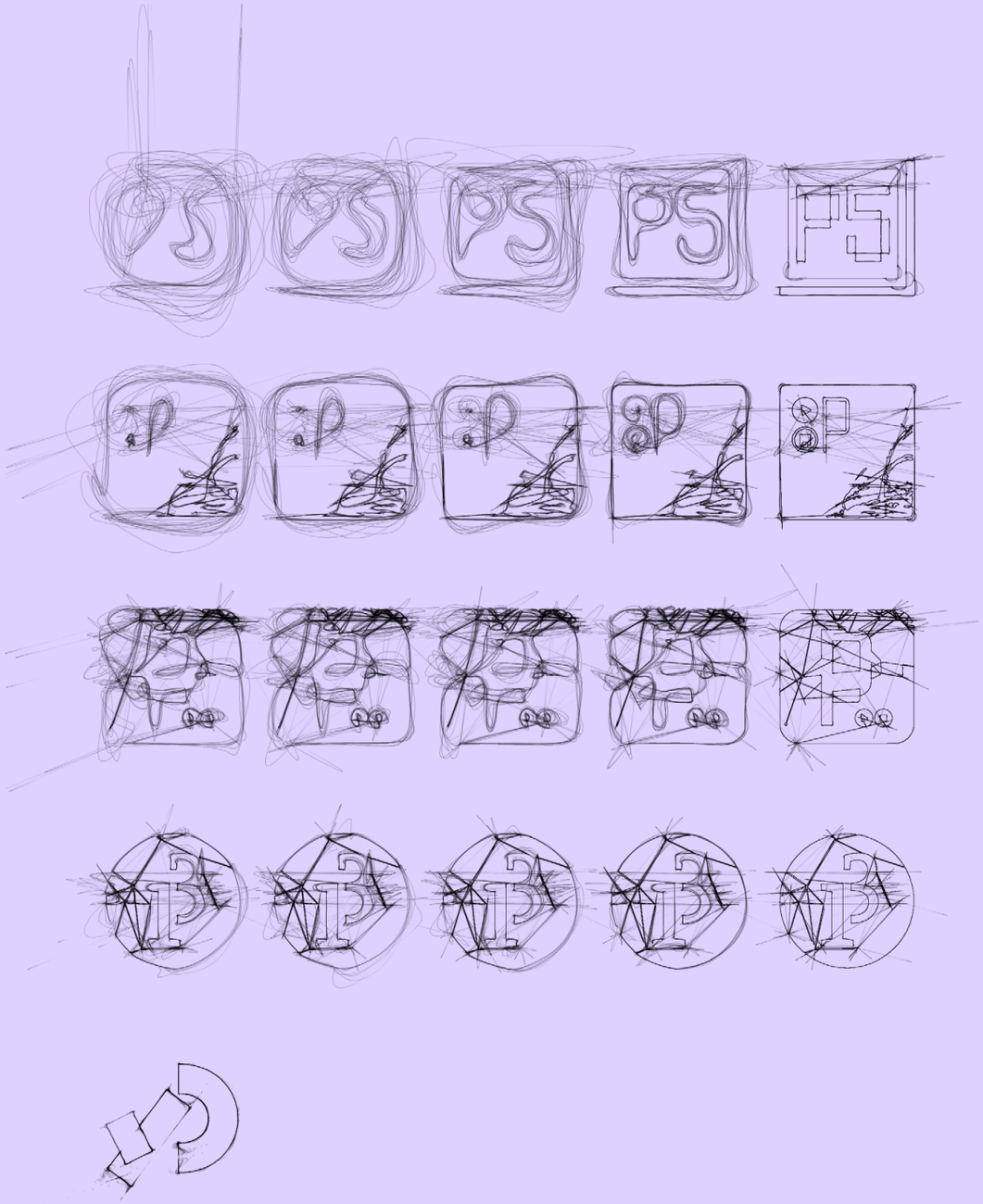





```
function setup() {
  createCanvas(2019, 2021);
  background(34);
}
function draw() {

  私とprocessingとの出会いは2018年です。
  当時私は大学の職員として働いていました。
  ある時、先生がコードで絵を描いている様子を見せてくれました。
  私は「この人はハッカーかもしれない」と思いました。
  訳のわからない数字や英語をパソコンで打ち込んでる人は、全員そんなふうに見えていました。
  自分にとって別世界のものでした。

  しかし、授業を受ける学生が同じようなコードを打ち込んで高度な絵が描けるようになっていきました。
  その様子を見て「すごい！」と思うと同時に「この子達に出来るなら、私にも出来るんじゃないか？」
  と思ってしまいました。
  2019年3月6日、初めてコードで絵を描きました。
  「絶対に関わることのない別世界」に足を踏み入れました。

  processingは私にとって本当に難しい分野です。
  ですが、「毎日やると出来るようになる」ということを体験させてくれました。
  これは人生において、本当に大切なことでした。

  そしてもうひとつ、「自分が描いたものを人に見せる」ということが出来るようになりました。
  初めはとても恥ずかしく、勇気が必要でしたが、毎日Twitter(@marie_usako)に投稿しました。
  すると段々恥ずかしい気持ちが無くなっていきました。
  その内、「そういえば私は手でペンを持って、紙に絵を描くのが好きだった」ということを思い出して
  絵日記を描くようになり、それもTwitterに投稿するようになりました。
  processing(#マリエちゃんのp)もpicture diary(#マリエちゃんの絵日記)も、
  上手ではないけれど、私は世界中の人に自分の絵を見せることが出来るようになりました。
  それは私にとって、とってもすごいことです！！！！
  2年前のあの日、別世界に足を踏み入れて良かったです。

  最後になりましたが、20周年おめでとうございます。
  そして、私にアドバイスをくれたり刺激をくれる皆様、本当にありがとうございます。
  これからもよろしくお願い致します。
  marie.
  2021.9.30
  https://twitter.com/marie_usako
  https://openprocessing.org/user/152145
}
```









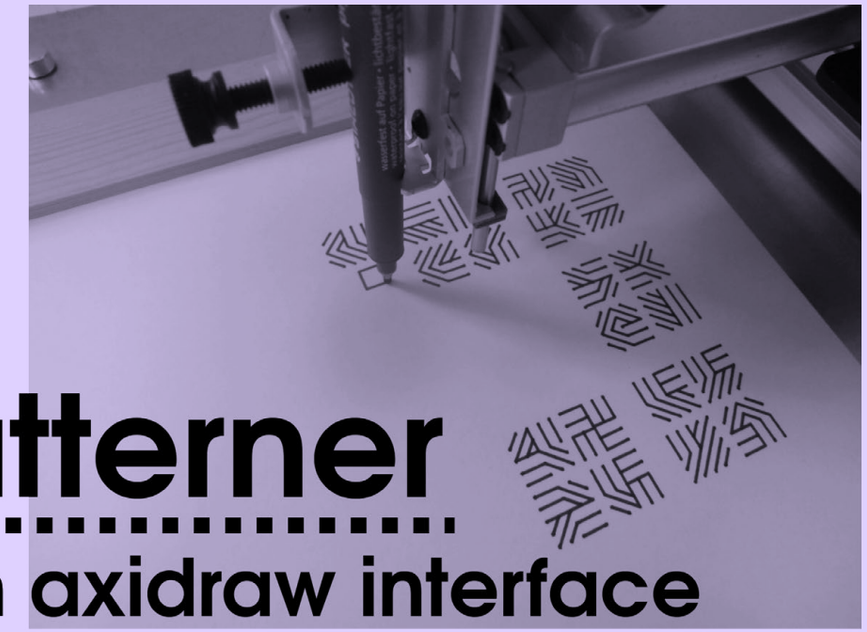
Growing up, I buried myself in novels about wizards and singersongwriter fan online forums. I felt weird and I didn't know how to tell other people about it. I wasn't even sure what kind of weird I was.

I first encountered Processing in an undergrad art class, where I learned I could connect with myself through making art. Art class assignments felt like someone saying "Here's some space, make something in it. We'll all look at it together when you're done." I was really into school.

Sharing the art I made felt like a safe way to share my feelings with others. Even if they didn't know what kind of weird I was either, our glimpses were enough.

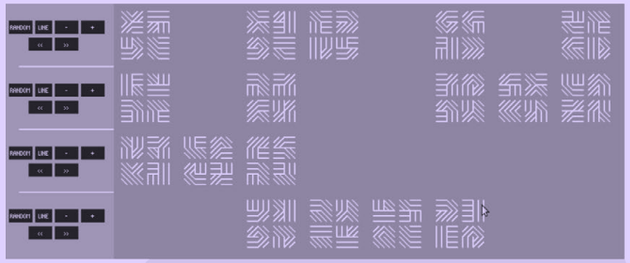
I contribute to p5.js because I want to support others in connecting with themselves and feeling seen. And because I want more friends. :)

made by matheplica



# Patterner

## an axidraw interface

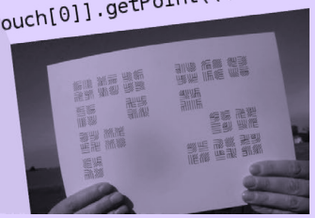


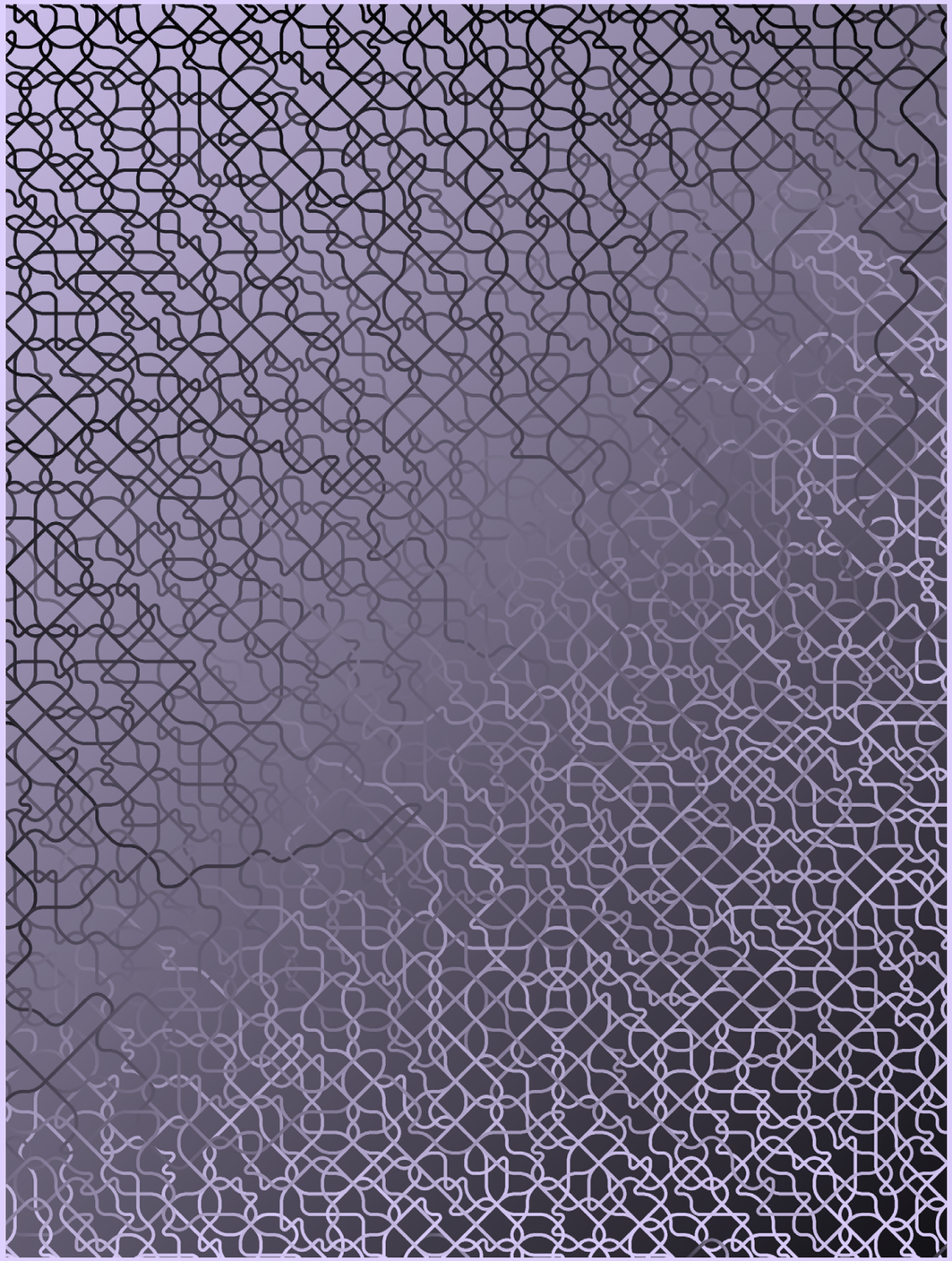
https://github.com/matheplica/patterner

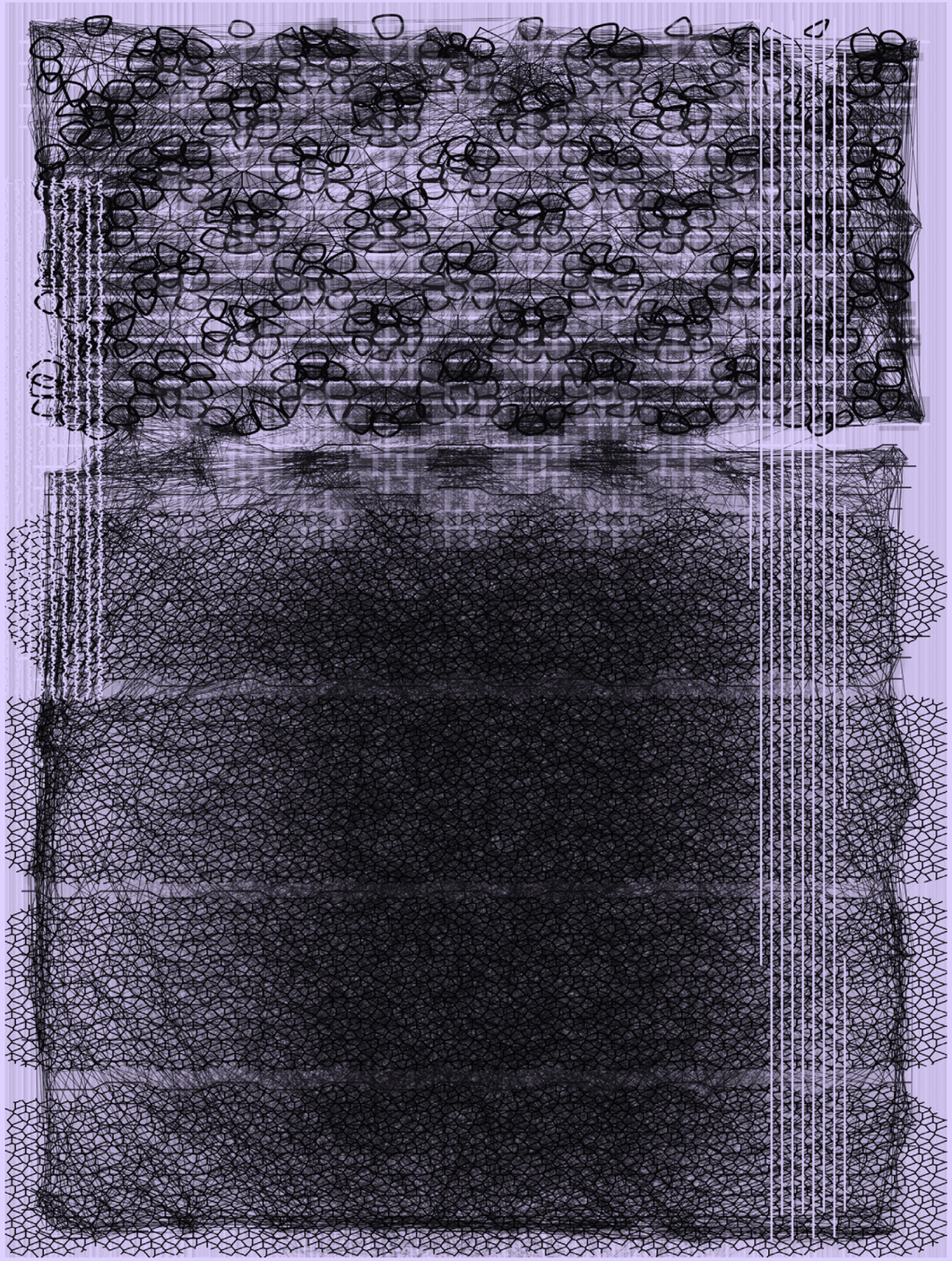
*I wanted to learn programming to use Processing 17 years ago. I was fascinated to can draw writting.*

*Here's a program fun to see in action because all path plotting are optimized. The function 'createWays()' who do this job was crazy to code.*

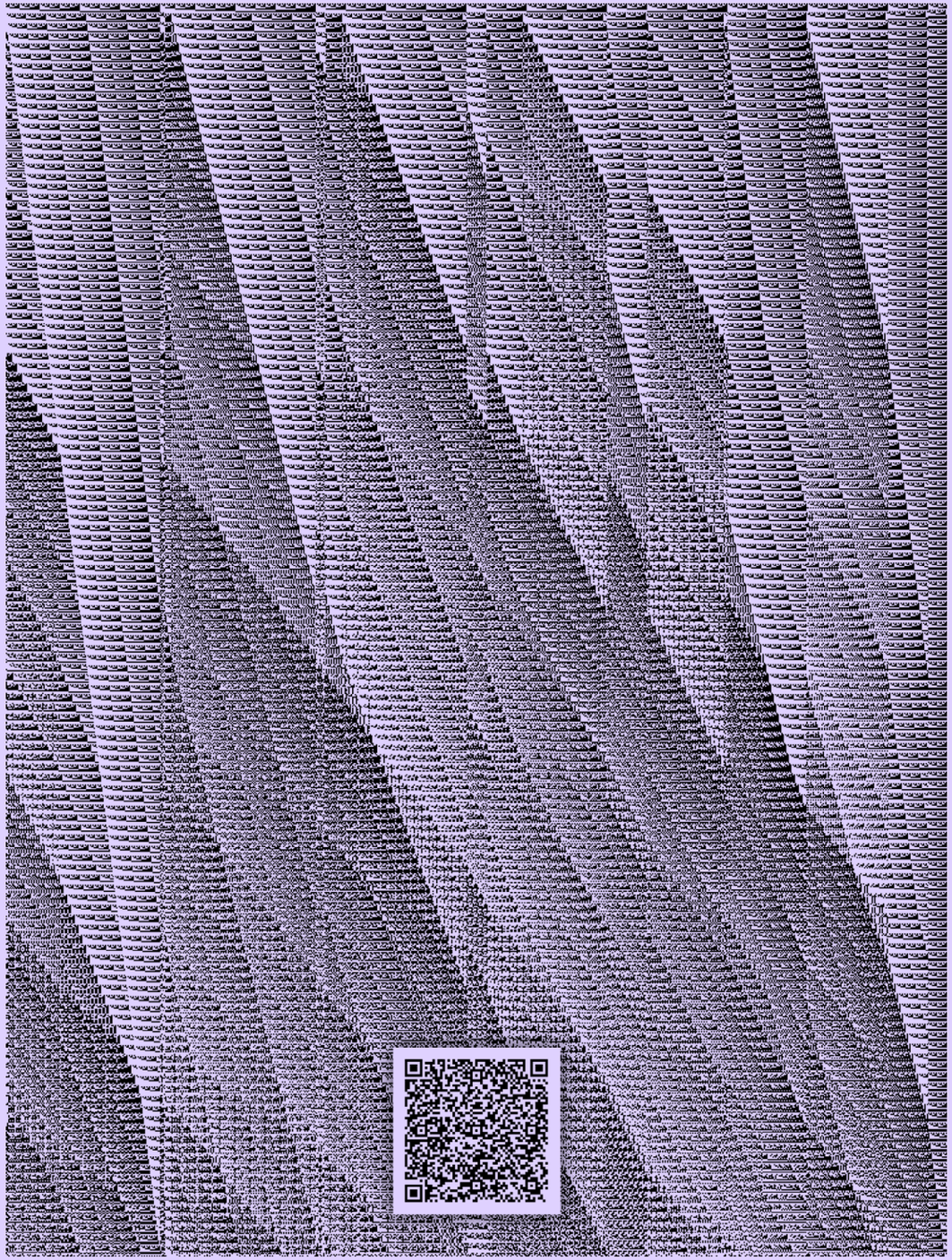
```
void createWays() {
    idSeg = 0;
    idPoint = 0;
    memSeg = 0;
    memPoint = 0;
    for (int i=0; i<12; i++) fullList[i] = 3;
    while (yetWork()) {
        if ((touch(idSeg, idPoint)==-1 && fullList[idSeg]>0)|| fullList[idSeg]==0) {
            ways.add(ways.size(), new Way(segments[idSeg].getPoint(idPoint)));
            idPoint = (++idPoint)%2;
            ways.get(ways.size()-1).addPoint(segments[idSeg].getPoint(idPoint));
            fullList[idSeg] = -1;
        while (touch(idSeg, idPoint)>-1) {
            ways.get(ways.size()-1).addPoint(segments[touch[0]].getPoint(((touch[1
            idSeg = touch[0];
            fullList[idSeg] = -1;
            idPoint = (touch[1]+1)%2;
        }
        idPoint = idPoint;
```

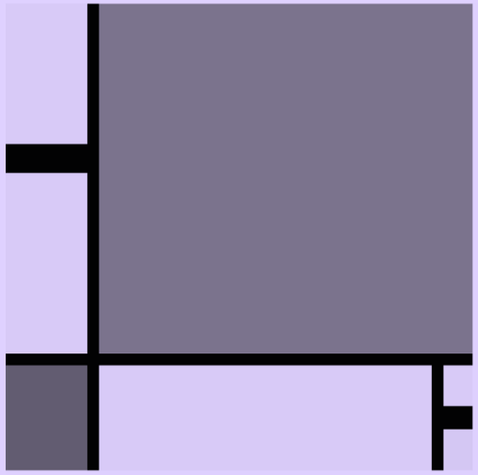












I began my creative coding journey in an art class over a decade ago. As an engineering student, I had come to view software as a tool to solve problems that mattered to me. But I had yet to make the connection between code and its limitless opportunities for self expression. Thank you, Gail, for showing me the light (and how to control it).

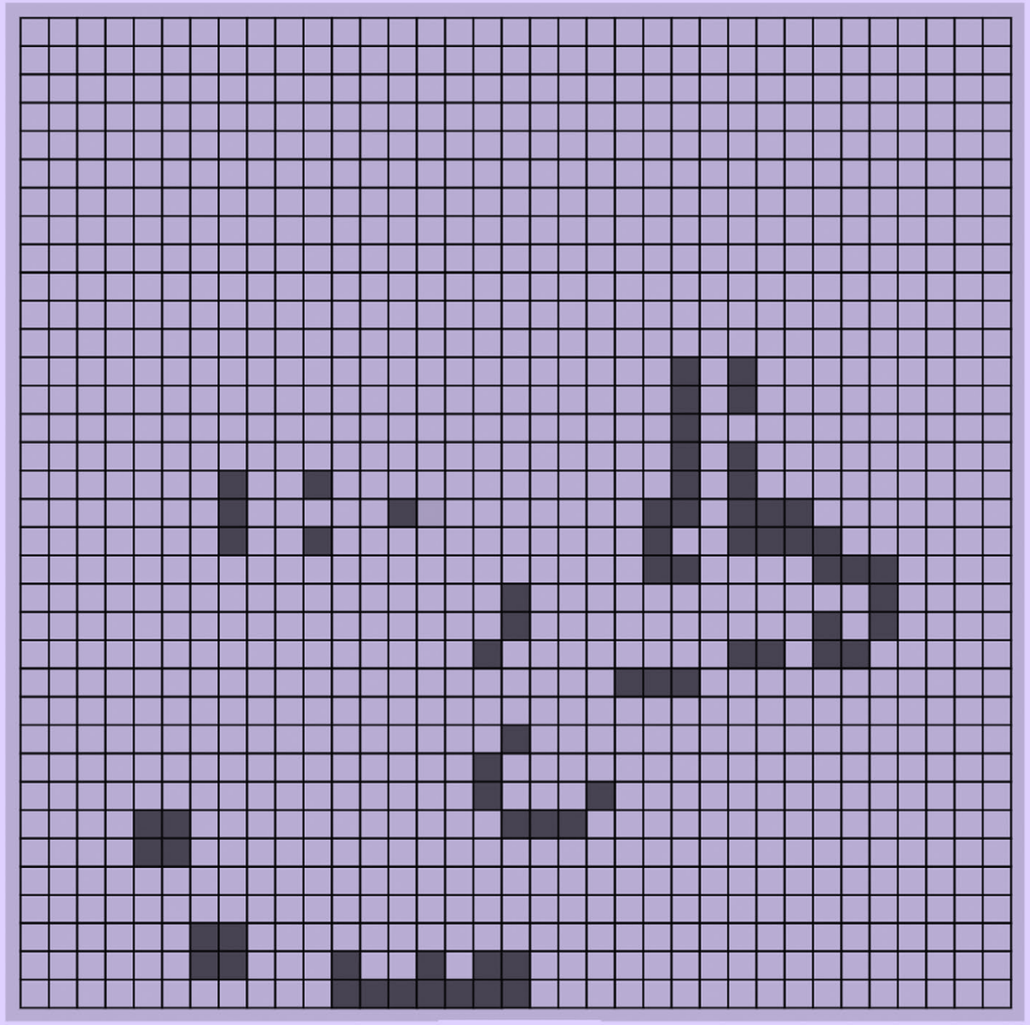
Cut to the present tense and I find myself in the unsettling position of doing what I love most each day: figuring out how to teach kids mathematics in a way that speaks to their hearts. The image above, a riff on Piet Mondrian's "Composition II", comes from an exercise wherein students abstract and abstraction with linear inequalities.

p5.js is the primary vehicle my students use to express their ideas visually and debug their own thinking. It's not overselling it to say that I've built my career on everybody's favorite software sketchbook, and I'm grateful to the Processing Community for all of the help and inspiration they've given me along the way. Thanks y'all!

With love,



## Game of Life Pong




Made with p5.js
joemckaystudio.com/gameoflifepong/

The Work of Art in the Age of Mechanical Reproduction





*Men of God, Men of Nature*, 2012 | Laleh Mehran

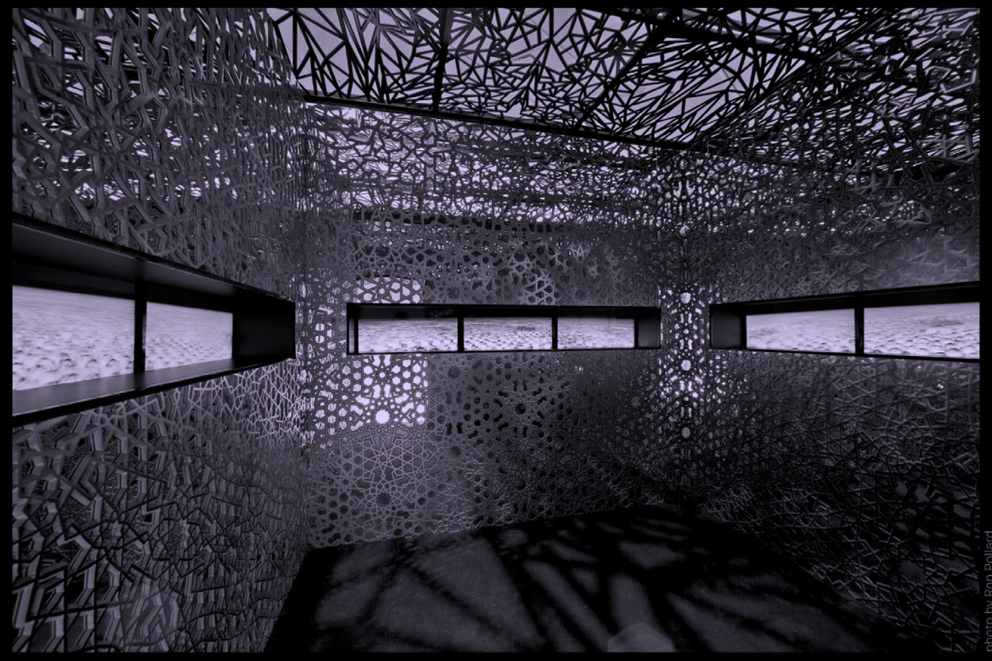
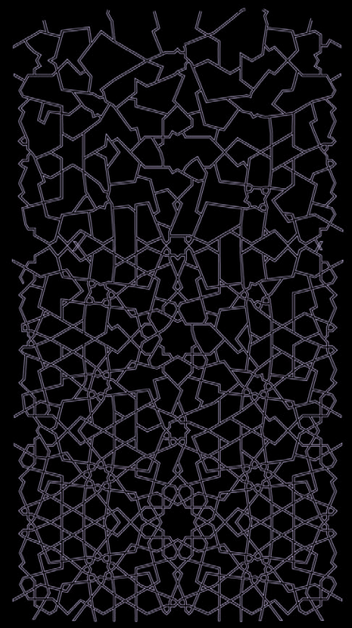
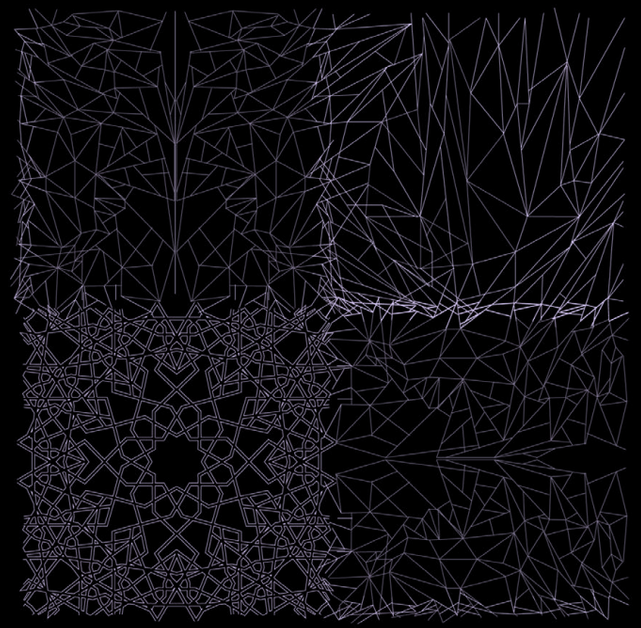


photo by Ron Pollard





How it started :
2018. Curiosity and Nature of Code.
It's been a slippery slope since.

How it's going :
2021. I am confident enough to call myself a New Media Artist. I teach Creative Coding at a Design School.

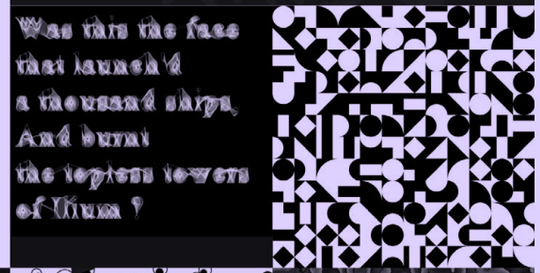
Exploring new areas within Math, Typography, Motion, Interaction..

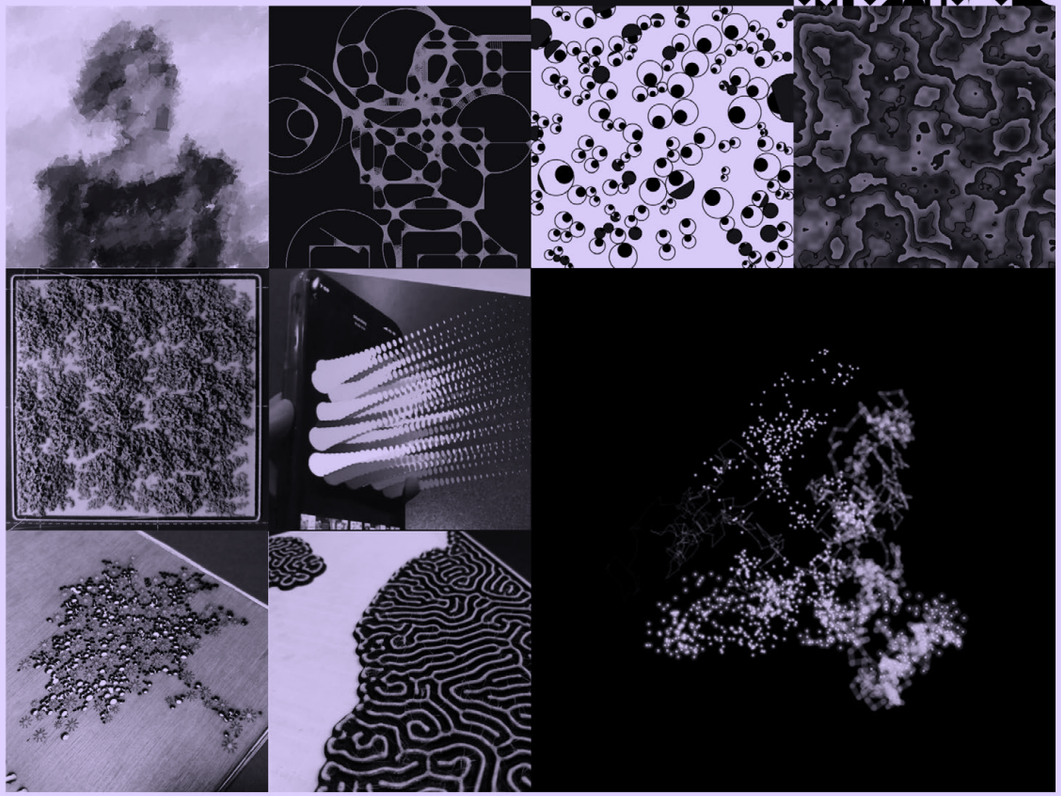
Bringing it back to the physical world using lasercutting, 3D printing, AR..

Flavouring it with my interests in poetry, photography, food and history..

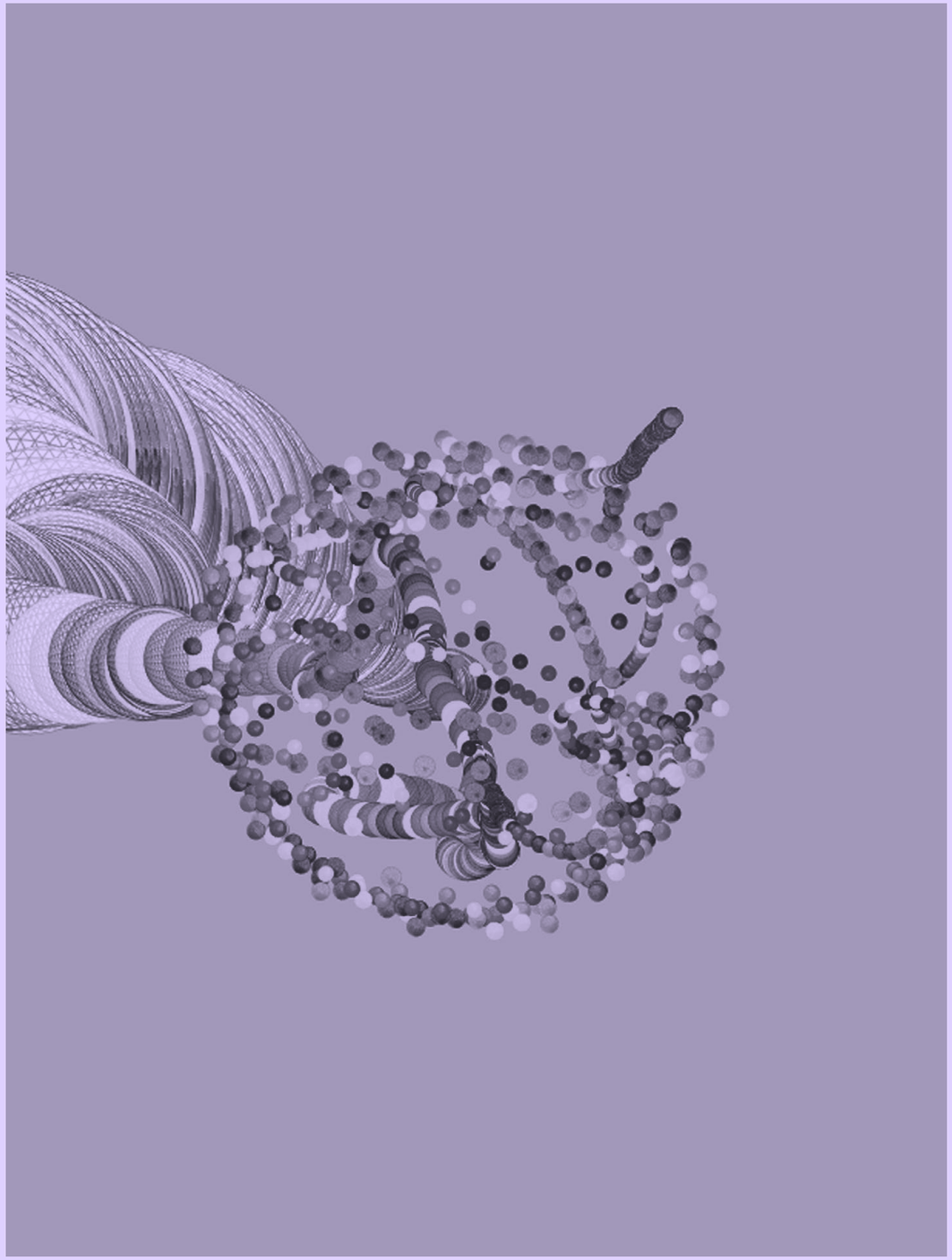
I am Jesal Mehta and working with Processing has changed my life.

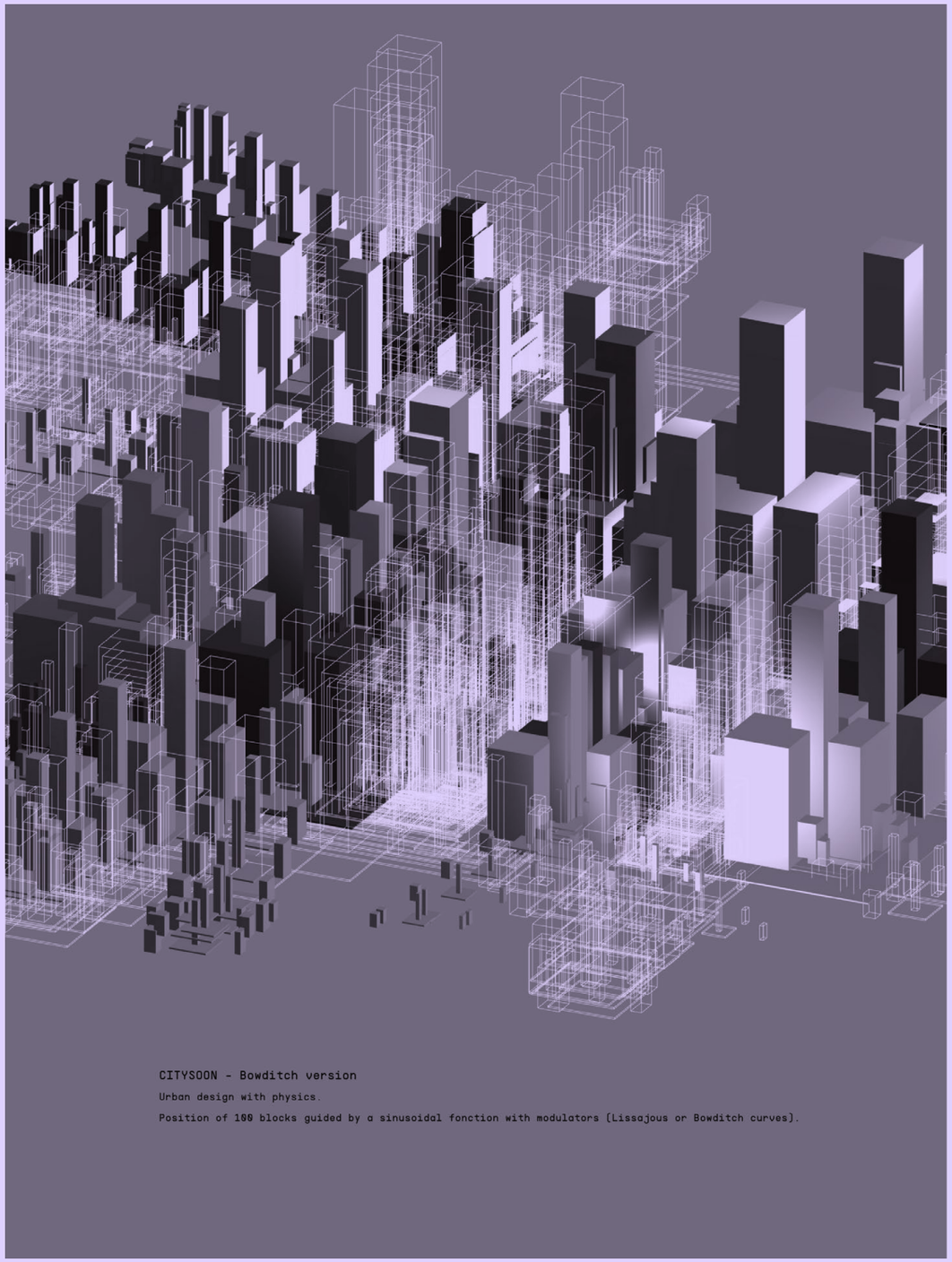
jesalmehta.com
https://linktr.ee/jesmehta



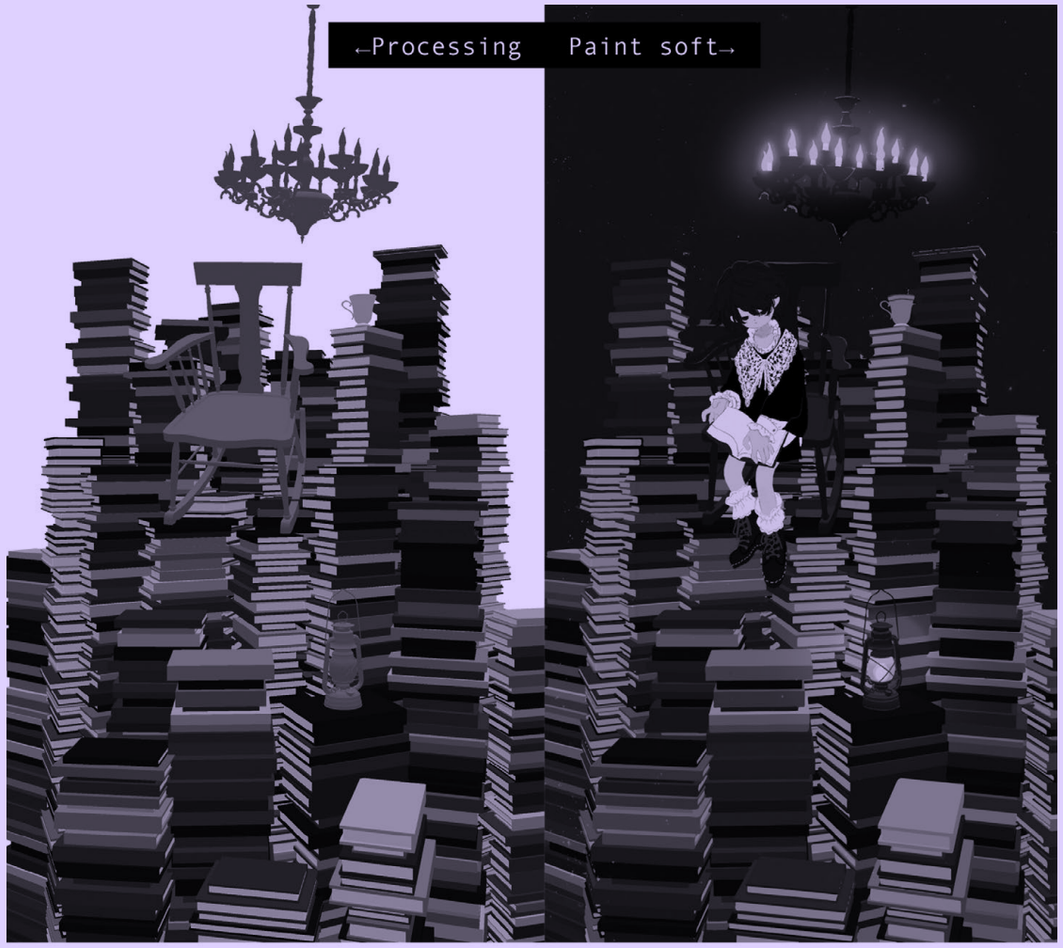











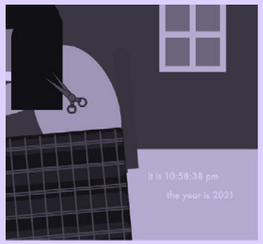



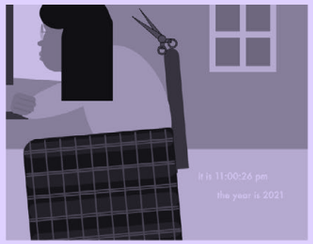

# Coding * Illustration

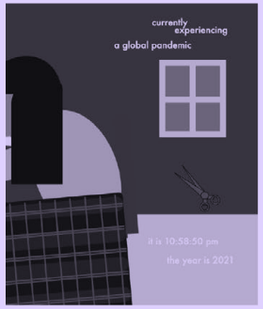
私にとって絵を描くのは大切なことです。でもゲームも好きです。
コードが書きたい、でも鉛筆や筆を持つのをやめることも絶対できない。
思いつきました、両方混ぜることにしました。

Drawing pictures is important to me.
I love Game, And Game graphics.
I want to write code using math.
I want to draw pictures with a brush.
I come up with the idea of “mixing”!

Books object
P3D,box(),color[]

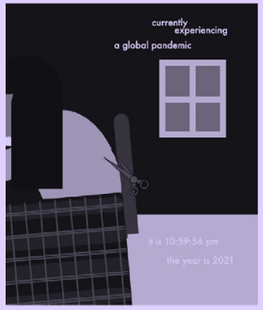


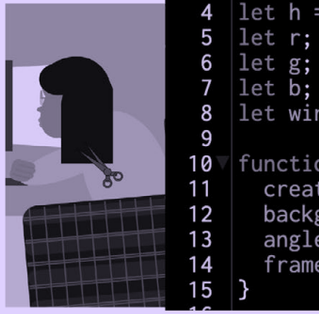











**currently experiencing pandemic time**

this piece is an experimental clock.

the screen flickers, bathing the expressionless face of a figure in light, and the digital clock at the bottom right is barely decipherable. the figure's hair grows, second by second.

2020 was the year that conventional, standardized time was exposed as an ever-present, arbitrary regulator. fluctuations in the body, self, and nature become re-centered as the loci of time.

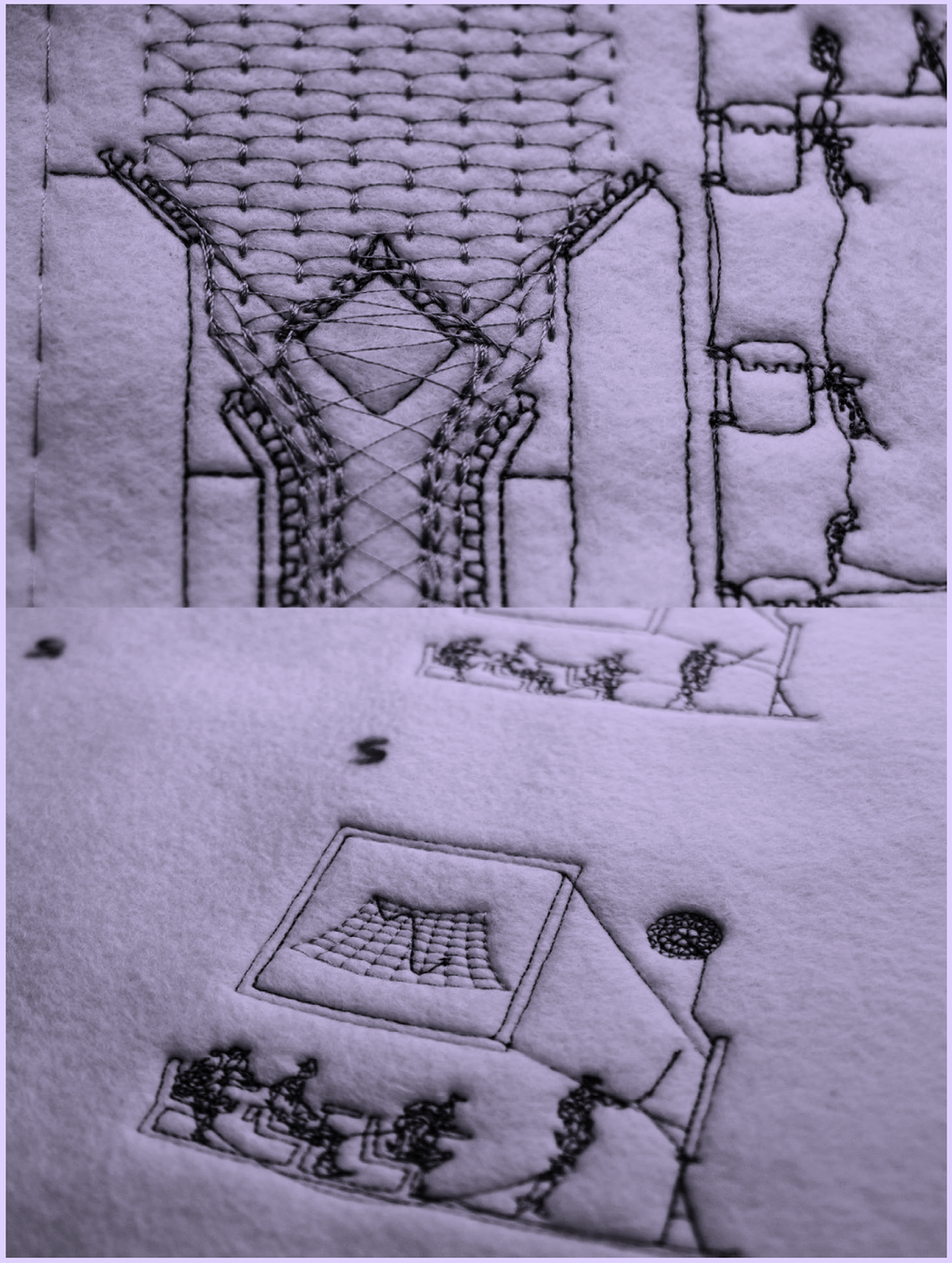
this p5.js-generated image shows the passage of time through consistent changes in hair growth, light, and clock digits, functioning as a reflection upon grief and artifice during and after the global COVID-19 pandemic.

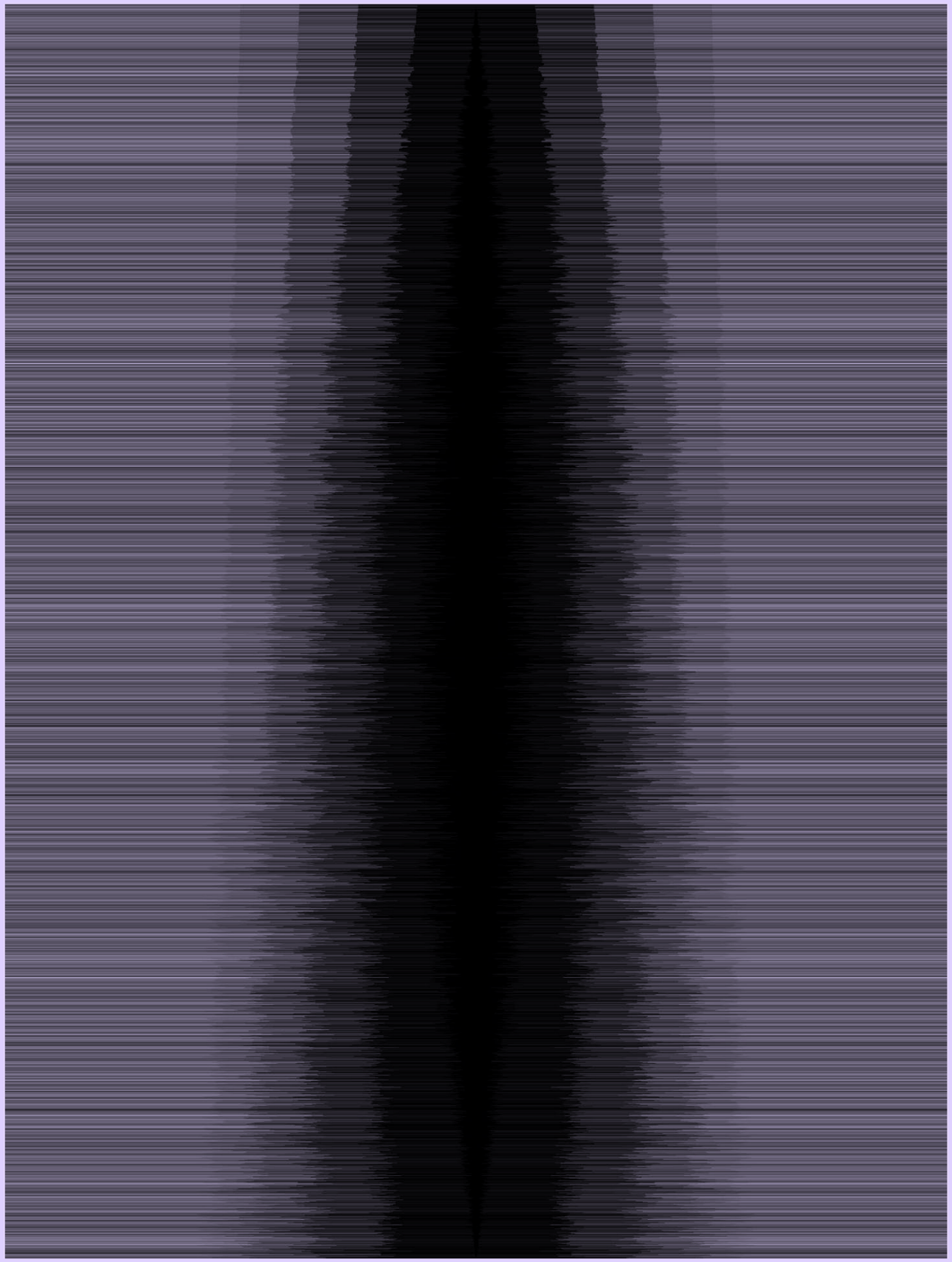
guinevere mesh (thedigitalguin.com / @thedigitalguin) programmed this timekeeper while at the parsons school of design (mfa in design + technology).

```
let ms = 0; //millis
let s = 0; // sec
let m = 0; // min
let h = 0; // hours
let r;
let g;
let b;
let winF; //light from window

function setup() {
  createCanvas(400, 400);
  background("#3F4152");
  angleMode(DEGREES);
  frameRate(10);
}
```

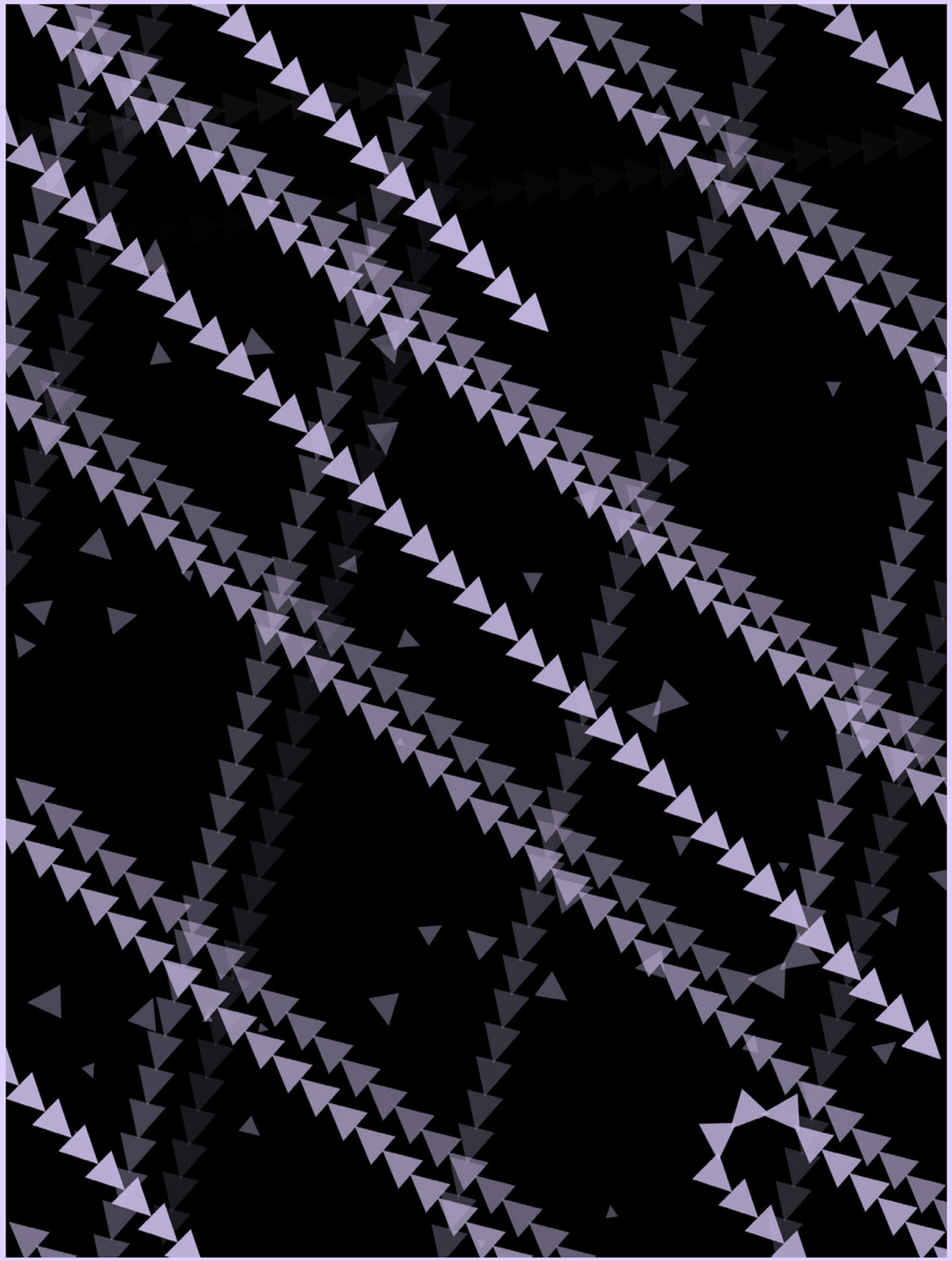
to view the code, visit:
https://editor.p5js.org/guinmesh/sketches/BEhjcFzXz







SANDRO MICCOLI CON 351 666



ANDREA MISURACA / @INSTAMISU CON 352 667

## Congratulations Processing on your 20th Anniversary!

Many Thanks to the Processing Community and leaders, especially Ben Fry and Casey Reas for starting it all!

Processing is a very helpful tool for me to explore my interests in photography and the Arts. I have been using Processing since 2010 starting with version 2. The many improvements over the years have certainly helped expand the range of projects where I could apply Processing.

I really enjoy writing code and Processing helps me satisfy my craving. I started coding in **1963** beginning with FORTRAN using punched card input. Thankfully we are much more productive today with the many tools we have like Processing. It helps that we can write code in Processing that runs on multiple hardware and software platforms for building or developing our artistic expressions.

Andy Modla
Software Engineer and 2D/3D Photographer

Github: @ajavamind

Here is a link to a Github page where I have links to my public Processing projects.
https://github.com/ajavamind/ProcessingSketches

These are my blogs links that have some postings related to Processing:
https://andymodla3dvr.blogspot.com/
https://andymodlaphotography.blogspot.com/
https://rcastudio2videogame.blogspot.com/

My Instagram accounts for photo and video:
Instagram: @andy_modla_photography @tekla3d

Twitter @andymodla

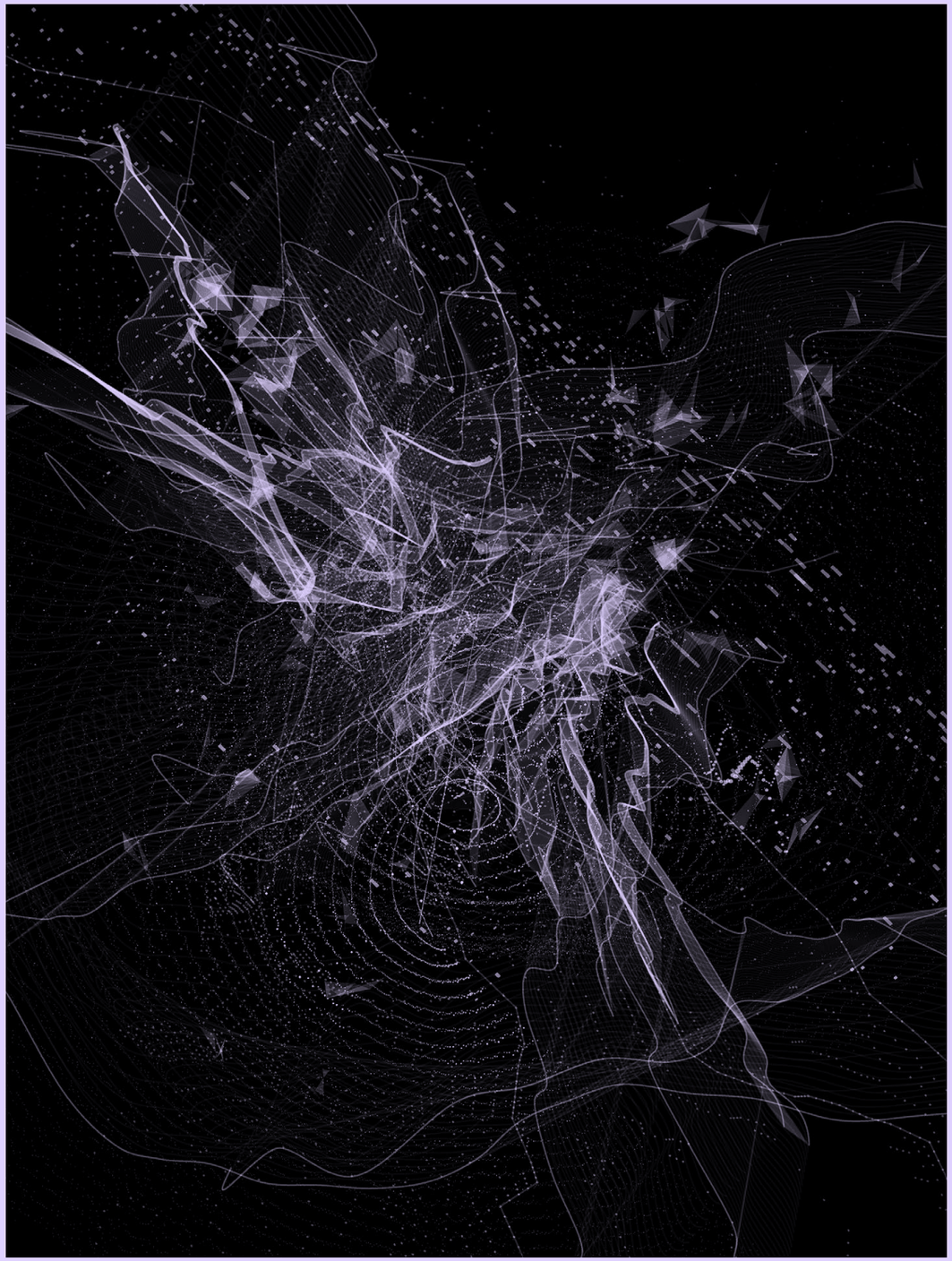
Here is a link to a YouTube video I made from frames generated with a Processing sketch. The frames were combined with a video editor and titles plus sound added.
My video and Processing code were inspired by the shredded performance artwork by the artist Banksy. His art “Girl with Balloon” was renamed “Love is in the Bin”. See article “Bansky Shredded Artwork Is Sold”, The Wall Street Journal, October 15, 2021, Page B6.

My shredded “Yellow Day Lily Flower” photo is now named “Buy Real Flowers For Me”.

| | |
|---|---|
| Shred Photo Art | https://youtu.be/zT58Xq-SjjI |

| | |
|---|---|
| Shred Photo Art 3D in Anaglyph Stereo | https://youtu.be/eS2jV-pJ1VI |





When I first discovered Processing in the autumn of 2007, I had no idea what a profound impact it would have on my career, my creative practice, indeed my life in general.

I've always loved mixing graphic design and coding. At first on my Amiga 500 at home in my teenage bedroom in the 1990s. Then as an apprentice in the design department of a printing company. Around the turn of the millennium I worked as a "New Media Designer" at an advertising agency, uploading table-based "business card" websites via crackling 56K modems and producing CD-ROM kiosks with Flash and Director.

In 2007 I was hired as a teacher in the Interactive Design Program at The Danish School of Media and Journalism (DMJX). It was here that I first encountered Processing. At that time it was a piece of beta software in revision 0135. It was love at first sight. Just working with a development environment where programs were called "sketches" and stored in a "sketchbook" was an indication that Processing was made BY designers FOR designers. I felt at home right away. A testament to my running love affair with Processing is my current sketchbook, which at the time of writing counts over 5000 sketches.

It didn't take long for me to put Processing on the curriculum. Selfishly because it was of interest to me, but primarily because the scholars of the day were stressing the need for programming and computational thinking to make its way into design education. I agreed. I thought it was narrow-minded to only teach students to be USERS of commercial software. I also wanted my students to become MAKERS of their own tools. To that end, Processing, with its low learning threshold and rapid iteration around the production of a visual output, was a perfect tool. And my students loved it from day one. So did I as a teacher. That split second when I can see that a student has grasped the idea of mixing design and programming, and suddenly sees their world of creative possibilities explode in a "big bang", is the moment I live for.

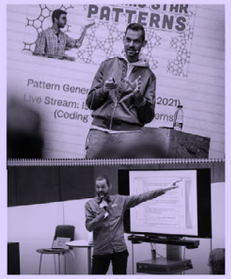
Since 2008, I have taught "Creative Coding" courses to about 700 students at various Danish design schools, and even more students abroad - both through physical workshops and remote teaching. Many of my students are now graduates and employed by many different Danish and international design agencies. The knowledge Processing has given them about how code can be used as a creative medium is now being applied in the design products my students create and which flow out into society. And I am genuinely happy about that.

Processing and p5 have also introduced a lot of pleasant people into my life. Over the years, I've traveled the world giving presentations about Processing and p5 at countless conferences, workshops, meetups, and Processing Community Days. Through this, I've met a lot of amazingly talented and pleasant people, all of whom share my passion for code and design. The vast international network and friendships I've made with Processing as a pivot point is a huge gift that keeps on giving day after day.

I have even been fortunate enough to have the opportunity to incorporate and apply my teaching experiences with Processing and p5 in both my Master's thesis, and subsequently also in my PhD. In my current job as Senior Associate Professor at DMJX, with creative code as my research area, I see it as my primary task to contribute to programming being embedded even more into the curriculum at design schools. Initially with an enthusiasm for the creative potential of programming, but subsequently - and just as importantly - also with a critical stance on the ethical, moral and societal impacts and consequences that are inevitably a by-product of software.

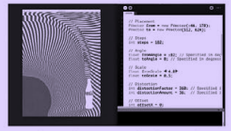
The brief on contributions to this Processing Community Catalog read, "What does being a part of the Processing community mean to you?" My answer is: "Gratitude". Gratitude that Ben Fry and Casey Reas, all Processing contributors, all the people their work stands on the shoulders of, and the entire community have enriched the world with a piece of open source software that empowers creative people to effortlessly express themselves through today's lingua franca: code.

Forever grateful,
/Stig
@Stixan · stigmollerhansen.dk



Speaking at design conferences about the wonders of Processing.


Teaching Processing remotely to students in Australia across ten time zones.


Doing live coding streams for the people in the community is something I would love to devote more of my time to do.

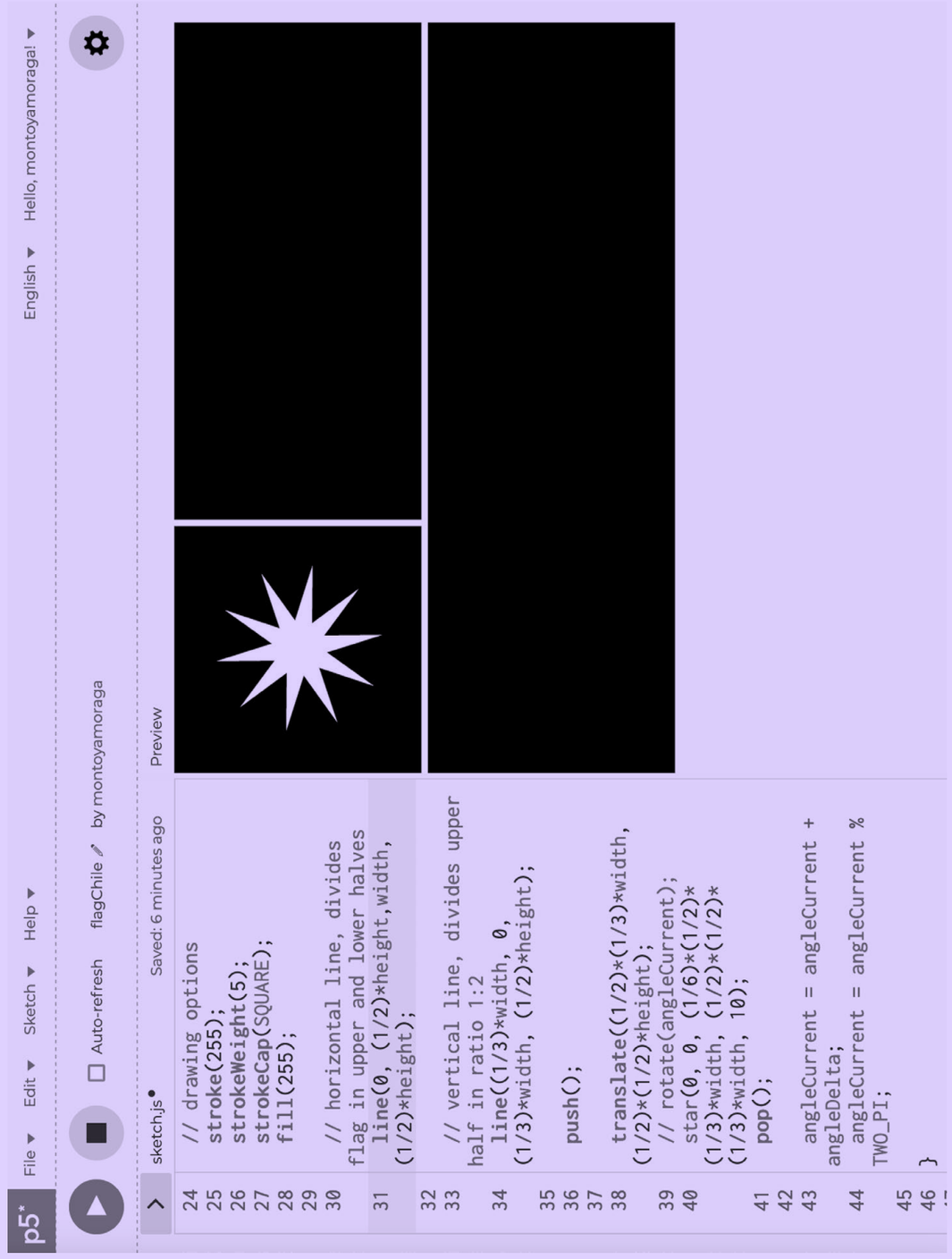
Three years in a row I've co-organized Processing Community Day Copenhagen.

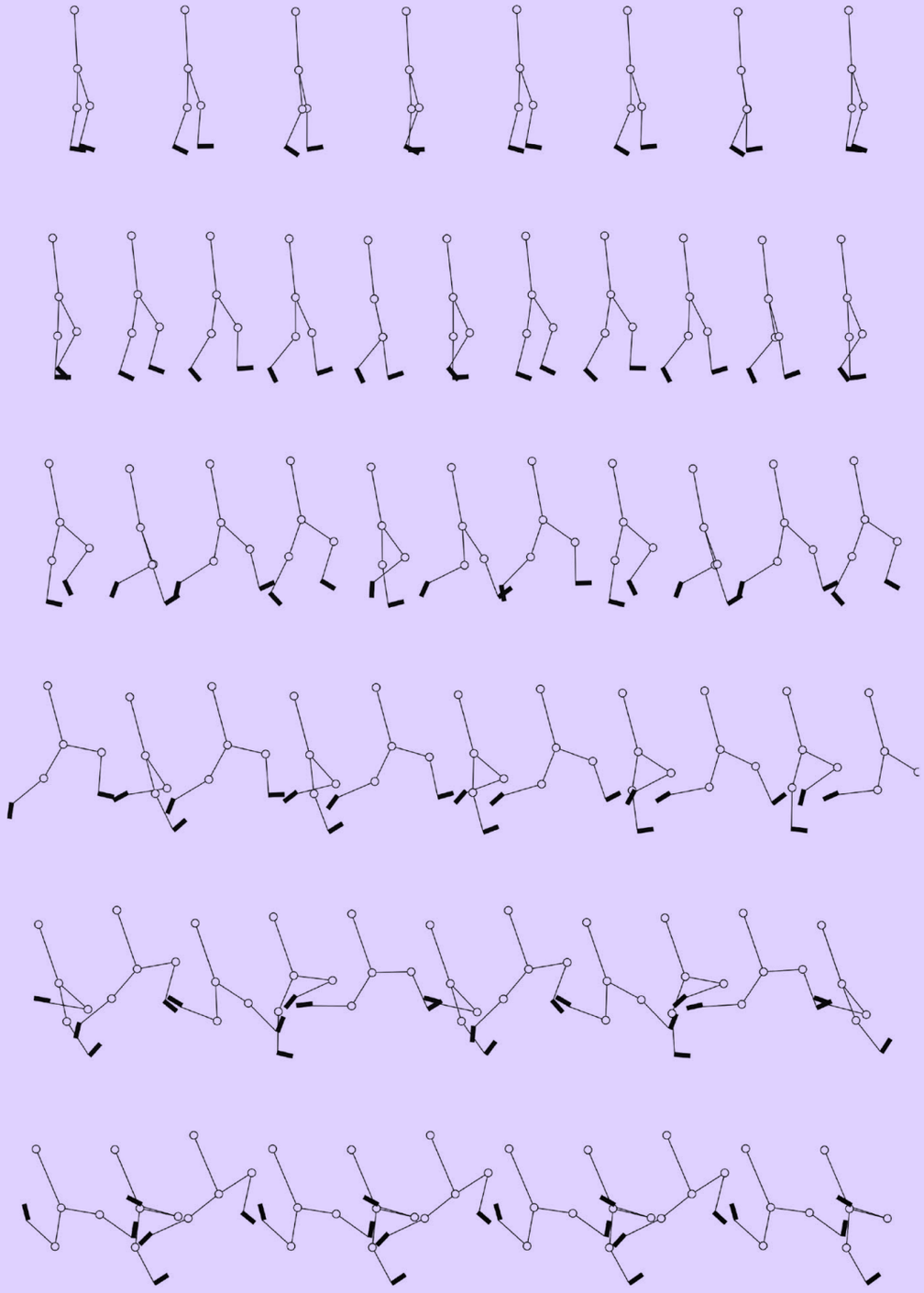
A gem in my library of coding books: "Getting Started with Processing" with a personal greeting from Casey and Ben.

Once I succeded in convincing a design magazine to devote an entire issue to the theme "Design <3 Code".

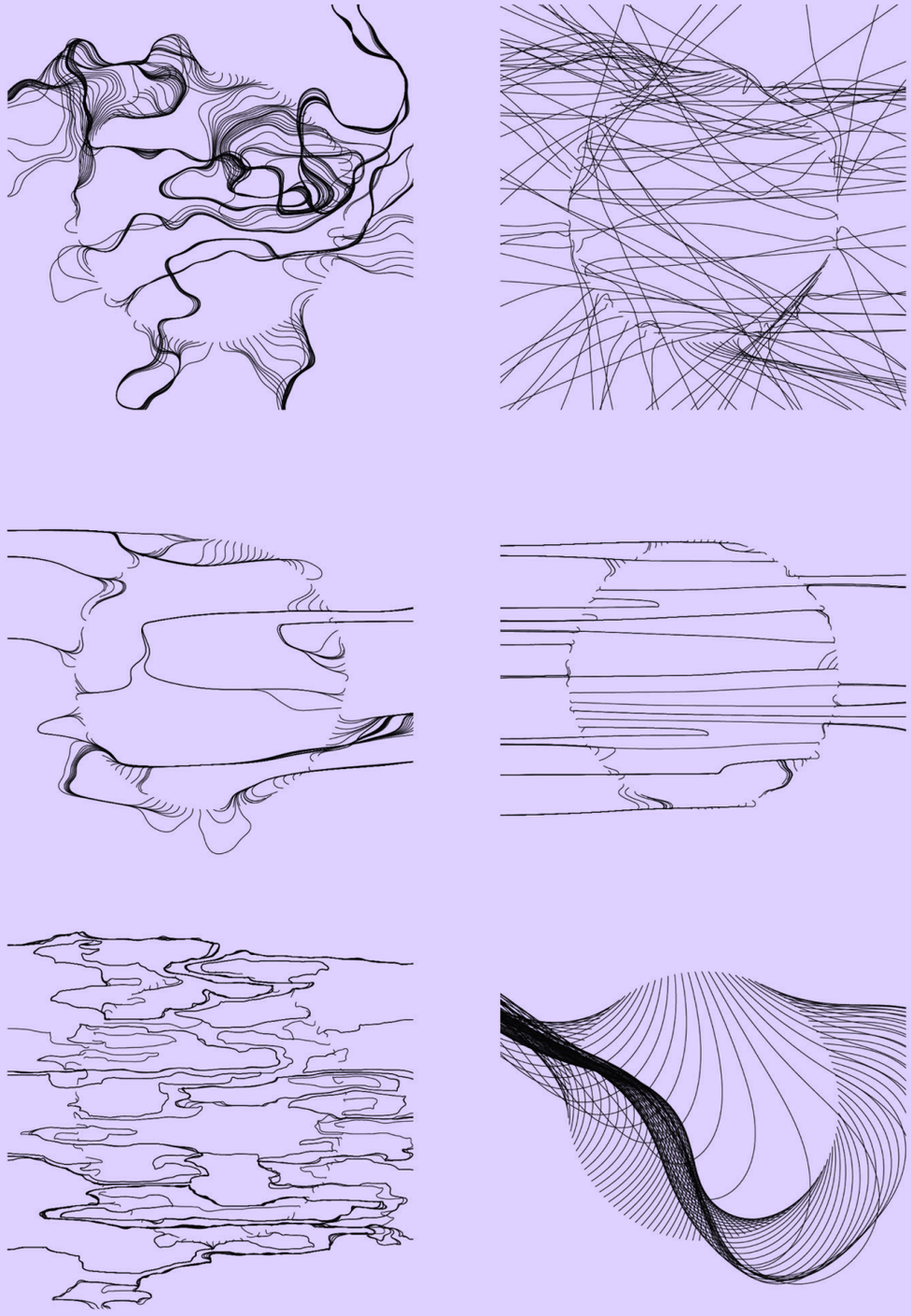
Resprayed my bike and gave it a set of "void setup(){}" and "void draw(){}" decals.












# VideoLab

by Eduardo Morais (2013-)

**What about using Processing to develop 'quick and dirty' educational software? I developed an application that teaches and lets students explore concepts of digital video technology. Video can be captured from the webcam or loaded from a file. My students could then experiment with:**

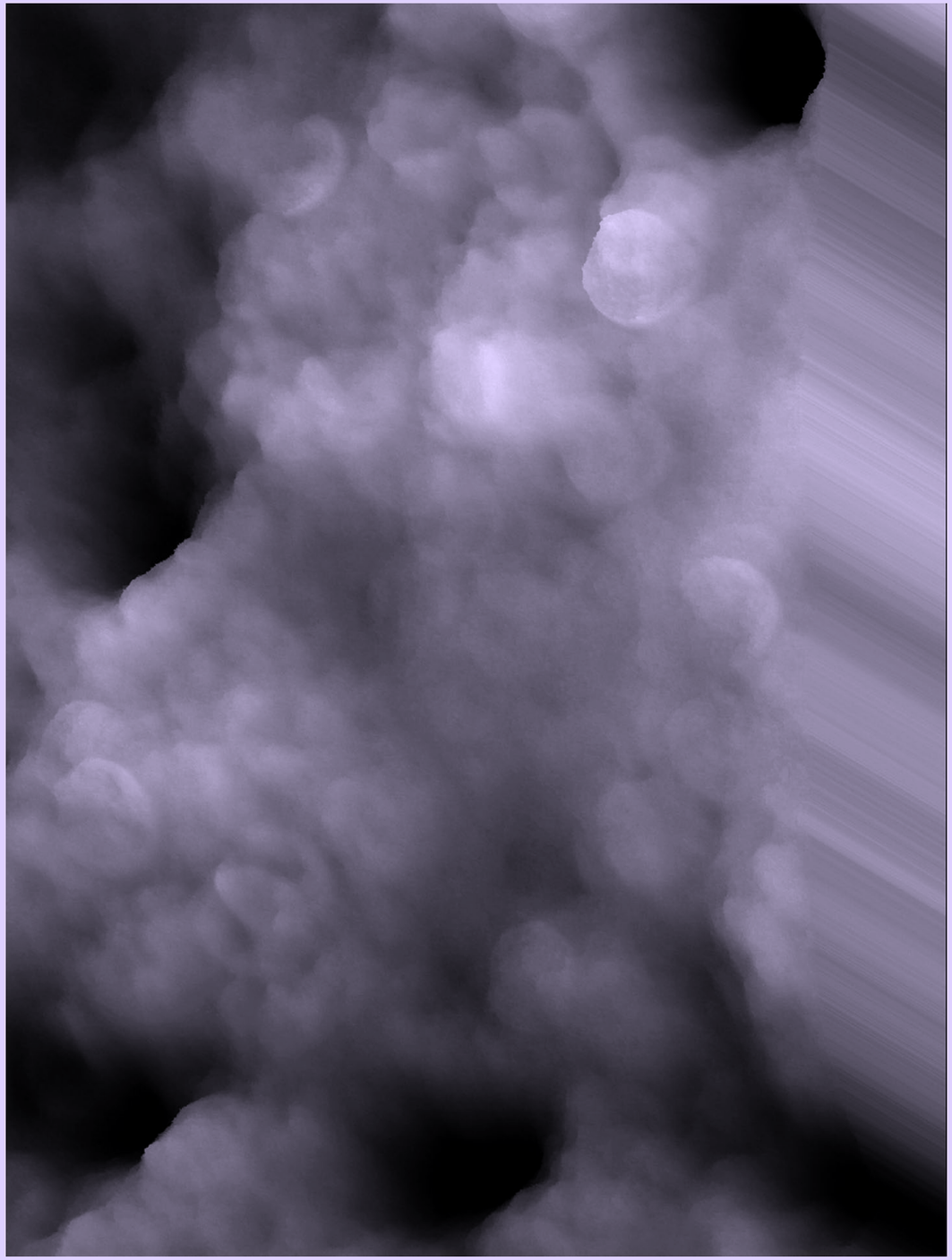
- **Picture aspect ratio;**
- **Progressive and interlaced scanning;**
- **Luminance, chrominance and subsampling;**
- **Color depth.**

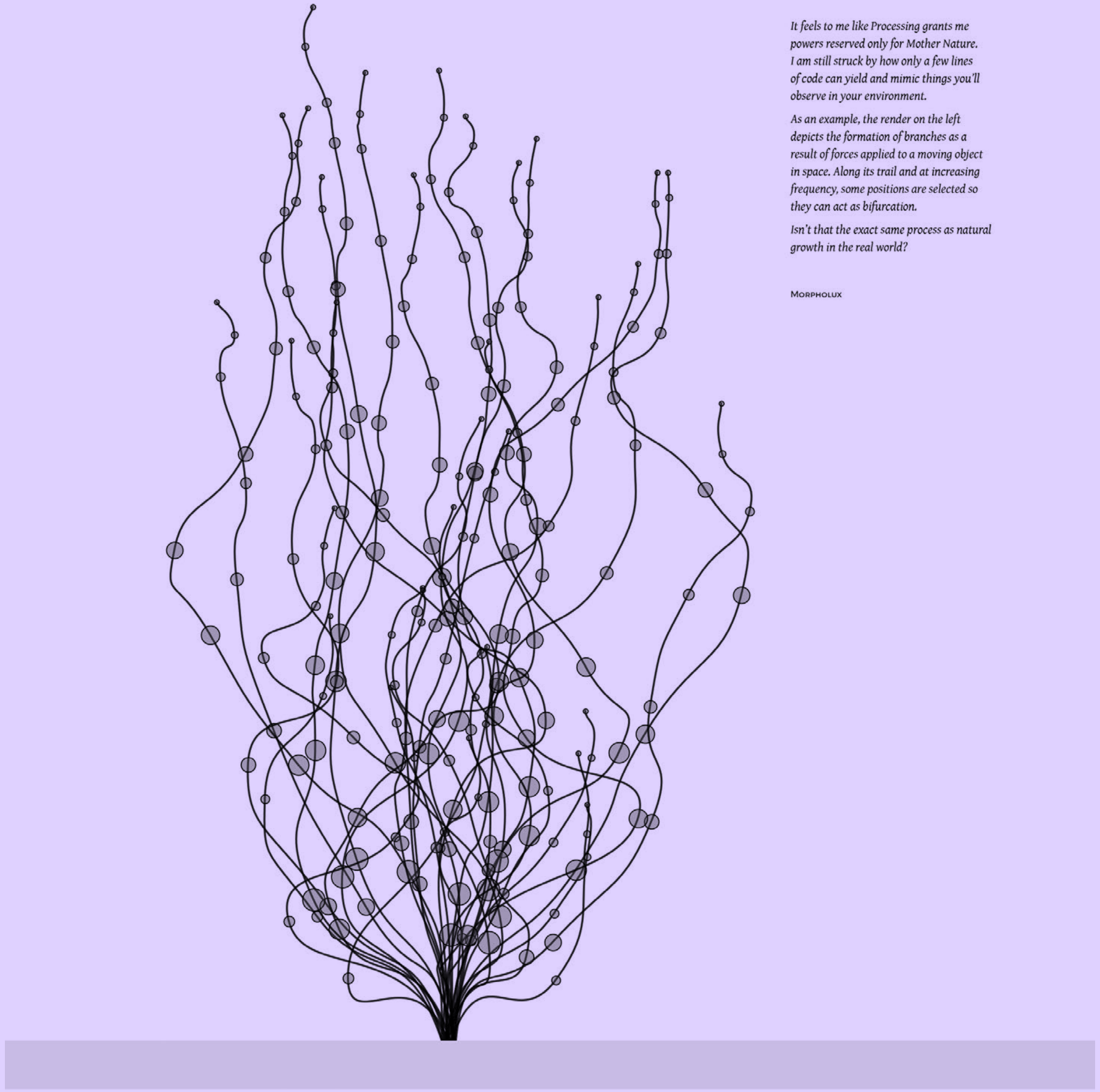
**The software also included onscreen tutorials for self-learning in the home.**

//github.com/edmorais/VideoLab









*It feels to me like Processing grants me powers reserved only for Mother Nature. I am still struck by how only a few lines of code can yield and mimic things you'll observe in your environment.*

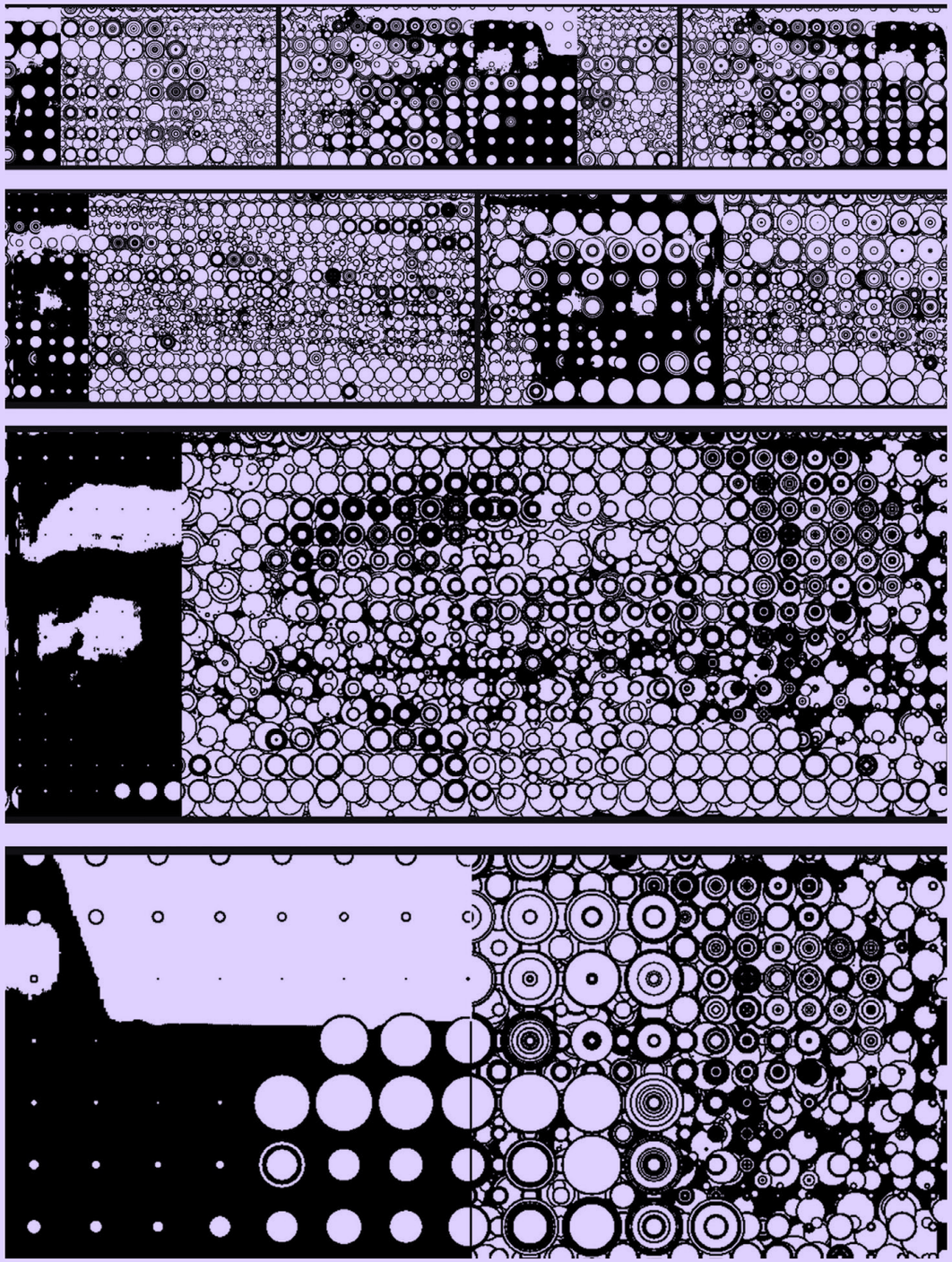
*As an example, the render on the left depicts the formation of branches as a result of forces applied to a moving object in space. Along its trail and at increasing frequency, some positions are selected so they can act as bifurcation.*

*Isn't that the exact same process as natural growth in the real world?*

Morpholux

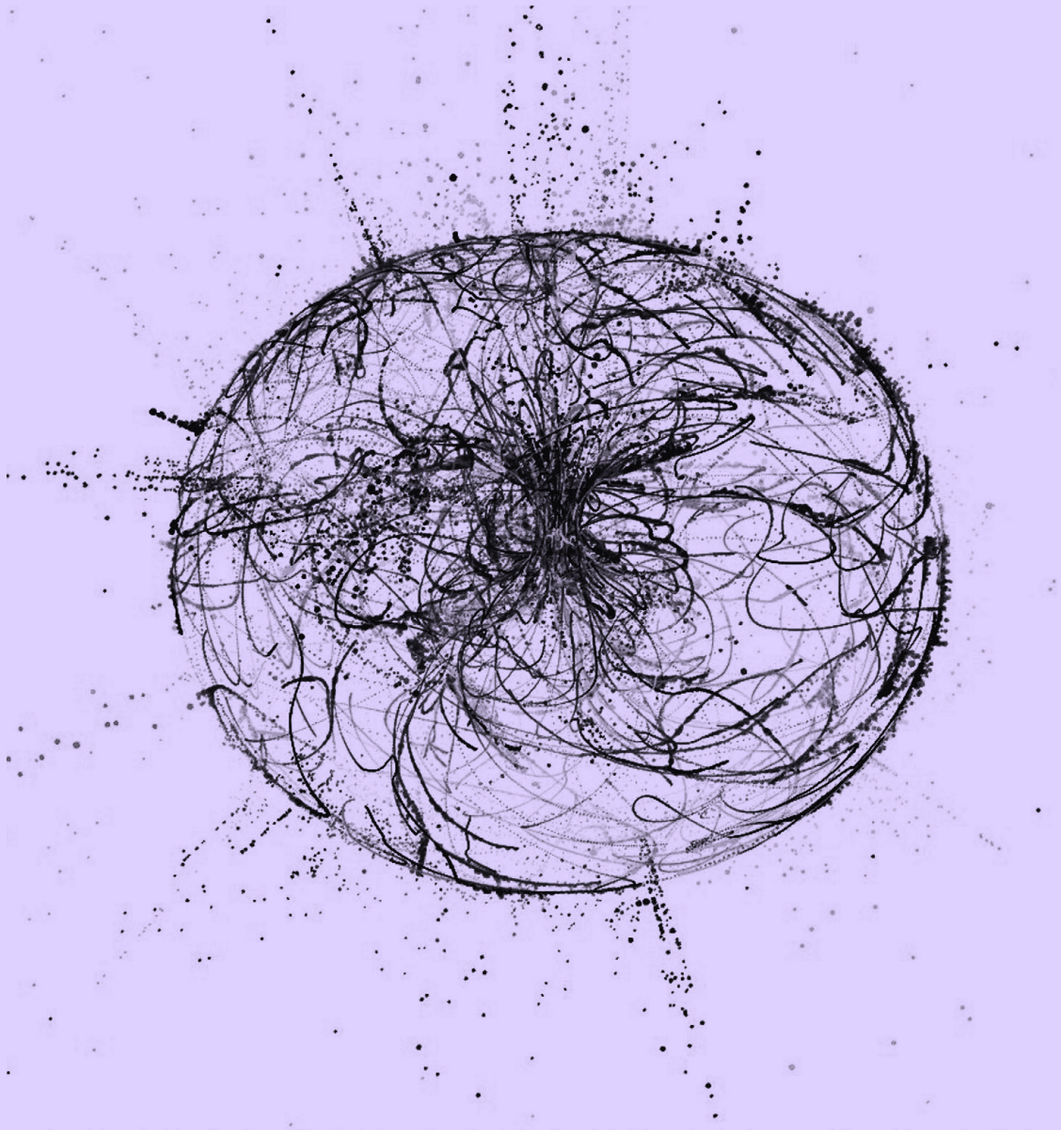
MORPHOLUX - JEAN-FRANÇOIS RENAUD

MR_SAMFIELD





# CELESTIAL POLLOCKS



I am interested in using Generative techniques to not only create art, but to understand some existing art in an algorithmic manner. Sometimes, unexpected resemblances emerge between unlikely processes and systems, which are moments of joyous serendipity, and open a door into emulating styles - *faux aesthetics* - as I like to call it.

This algorithm uses a piece of music as input, where the spectral modes are mapped onto a system of bodies following Newton's law of gravitation, which then paint out a Jackson Pollock inspired shape of splatters and trailing lines. Here's Erik Satie's Gymnopedie 1 made into into an abstract painting.

- Siddhartha Mukherjee

DECODINGKUNST

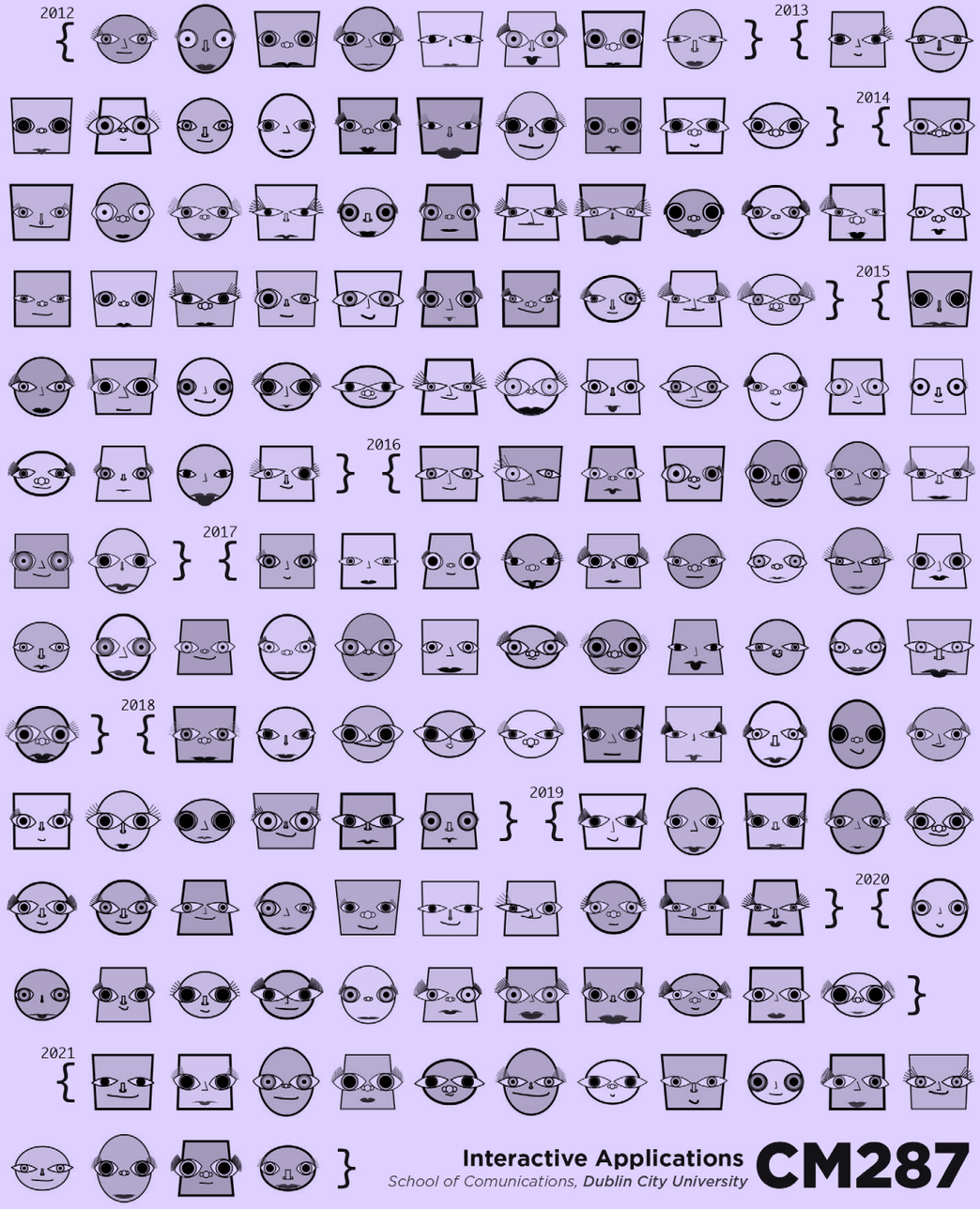
In 2019, the Prinsenhof museum in Netherlands, together with the *Inovatie* group, reached out to the Delft University of Technology (TU Delft) for a collaboration. The idea was to give a modern technology perspective on the classical paintings of Pieter de Hooch (a contemporary of Johannes Vermeer), as part of a retrospective exhibition on this Dutch master. Students following the "Algorithmic Art with Processing" course at "X" (the TU Delft Culture Center) took this up as a project, and came up with many interactive algorithmic art interpretations of the classical paintings, several of which became part of the exhibition (selected works below).

Clockwise: 1.) *Triangulation* by **Roger Willemsen**, 2.) *Electric Boogaloo* by **Jesse Nijdam** and 3.) *Dot Matrix* by **Siddhartha Mukherjee** (course instructor, Instagram @decodingkunst).

TUDelft

DECODINGKUNST



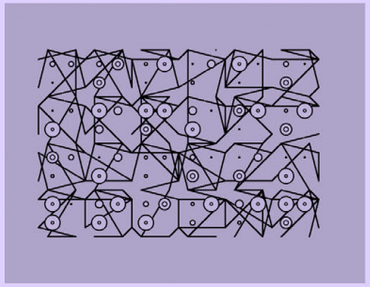

**Interactive Applications** **CM287**

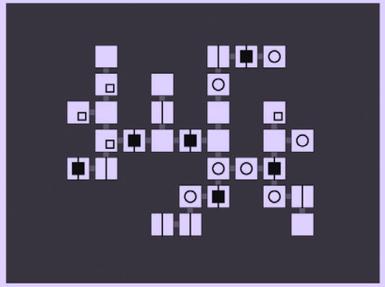
*School of Comunications,* ***Dublin City University***

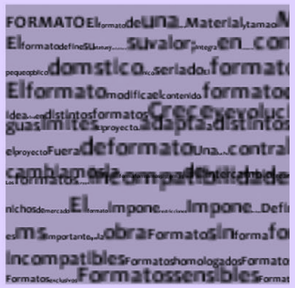
**Dónal Mulligan**
*@donalmulligan*

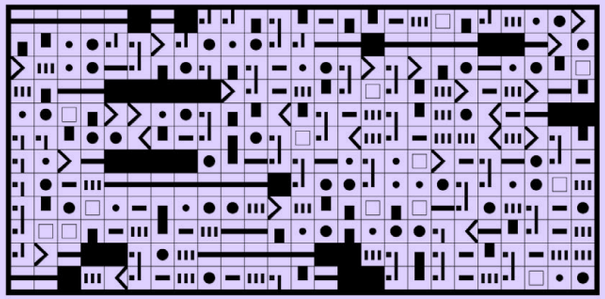
The faces on this page represent each of the students who have taken this module, on the *BSc in Multimedia* at DCU, since it began. The module uses *Processing* to teach coding and concepts for the creation of interactive media. Fittingly, these faces were generated using a *Processing 4* sketch, which took each student's ID number as a genetic input and created a unique face for each.

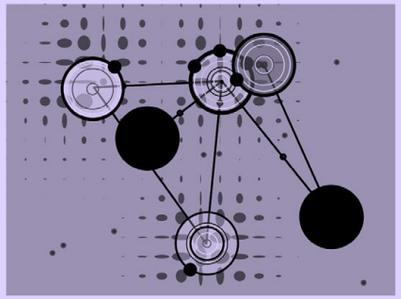
Jaime Munárriz

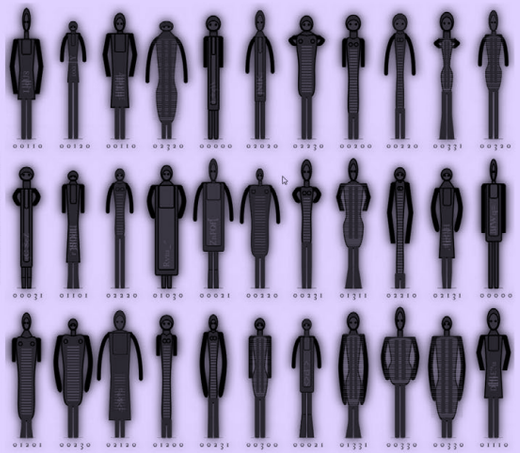


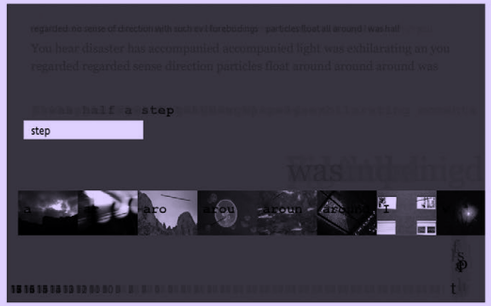


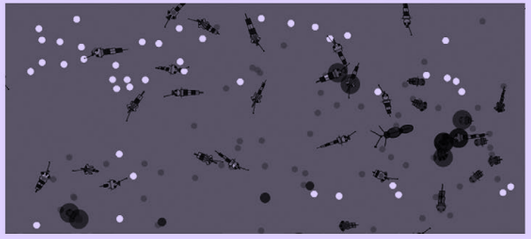








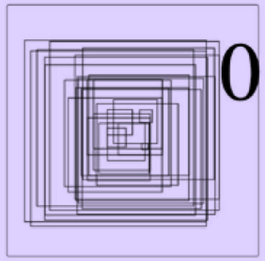





# Making the Thing that Makes the Thing

Recreating classical works in generative art & design with p5.js. A workshop conducted at Processing Community Day, Bangalore 2020.

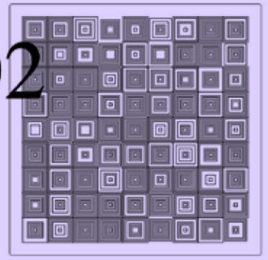
**Musings with Code**
musingswithcode.studio/generative-design-workshop
Ajith Ranka & Priti Pandurangan



01
Vera Molnár
*Untitled 1974 (Squares)*
Go to sketch

05
Sébastien Truchet
*Tiling (Triangles)*
Go to sketch

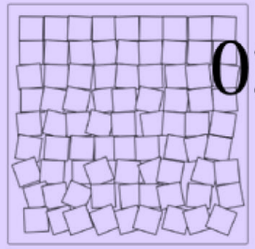
Vera Molnár
*Des Ordres 1974 (Grid)*
Go to sketch



02

C S Smith
*Tiling (Waves)*
Go to sketch

06



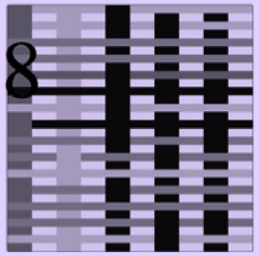
03
Georg Nees
*Schotter (Gravel), 1968*
Go to sketch

07
Piet Mondrian
*Mondrian Grid*
Go to sketch

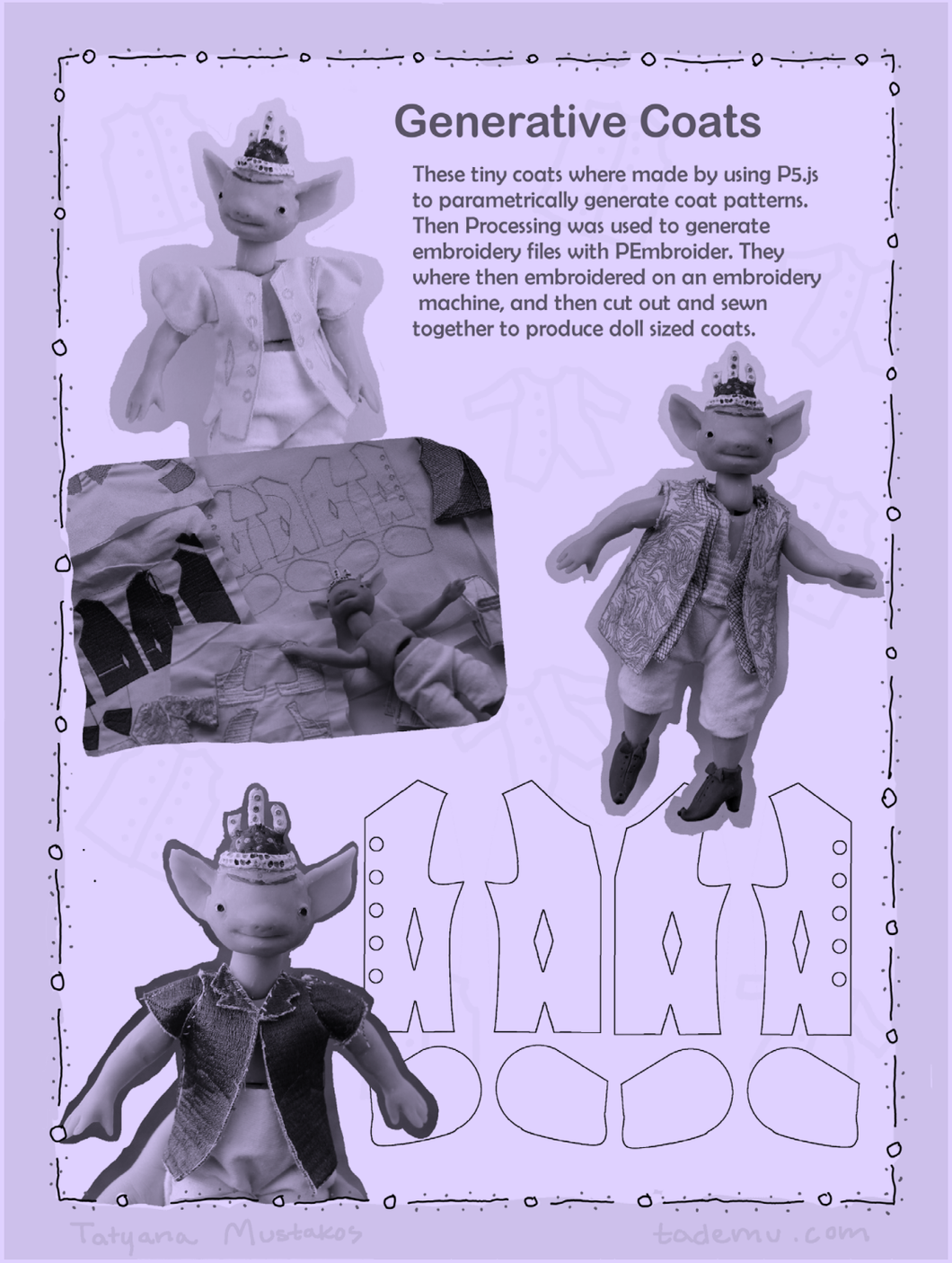
10PRNT( );
Go to sketch

04

Anni Albers
*Generative Weave*
Go to sketch



08

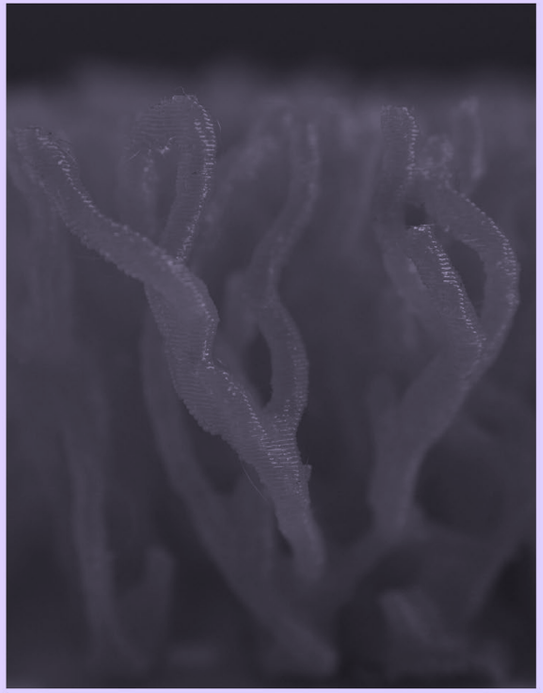


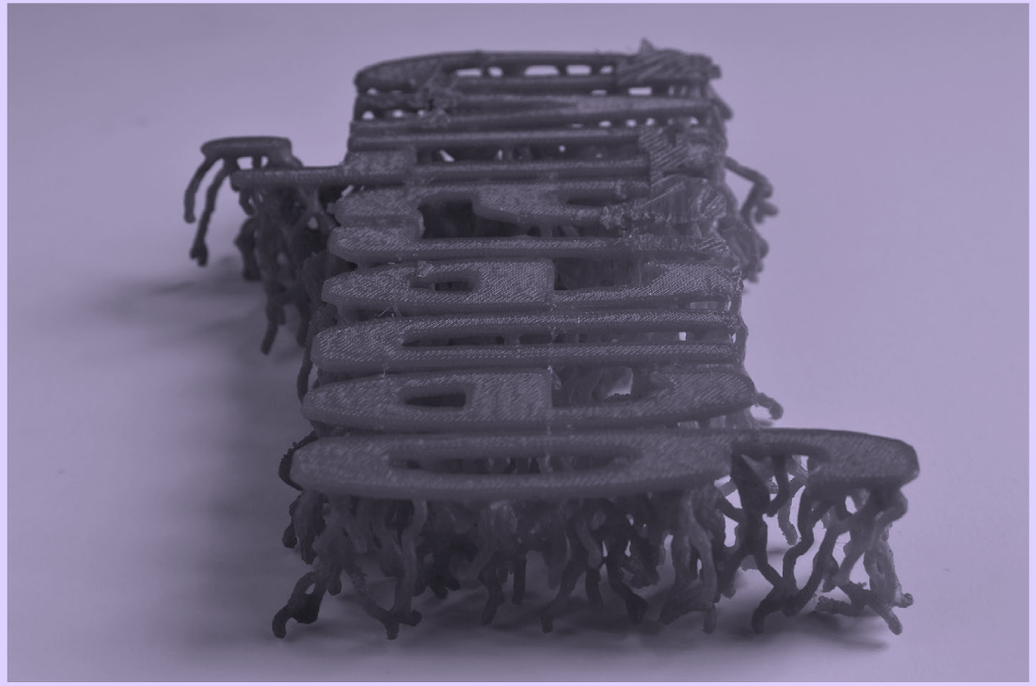


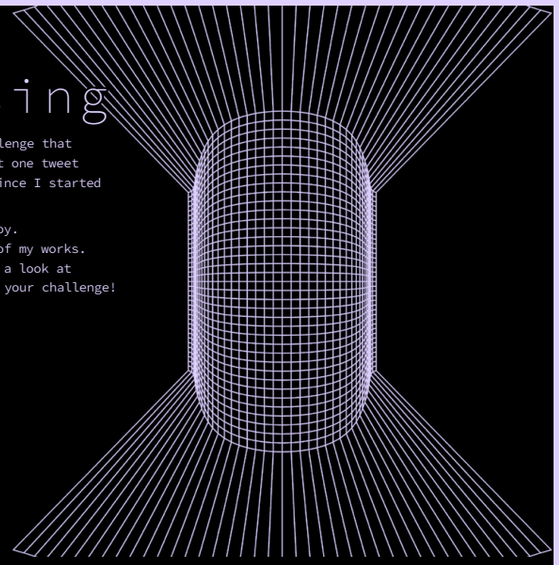

# Generativo

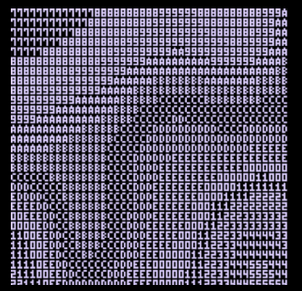
Coral like structure grown from the word “generativo” using diffusion limited aggregation. 3D print.

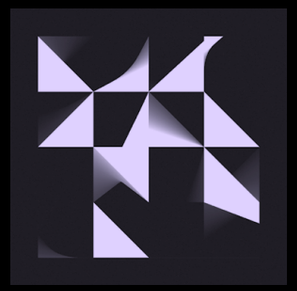
nacho cossio, 2015
@nacho_cossio

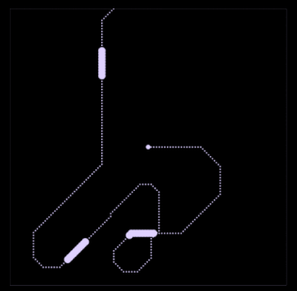


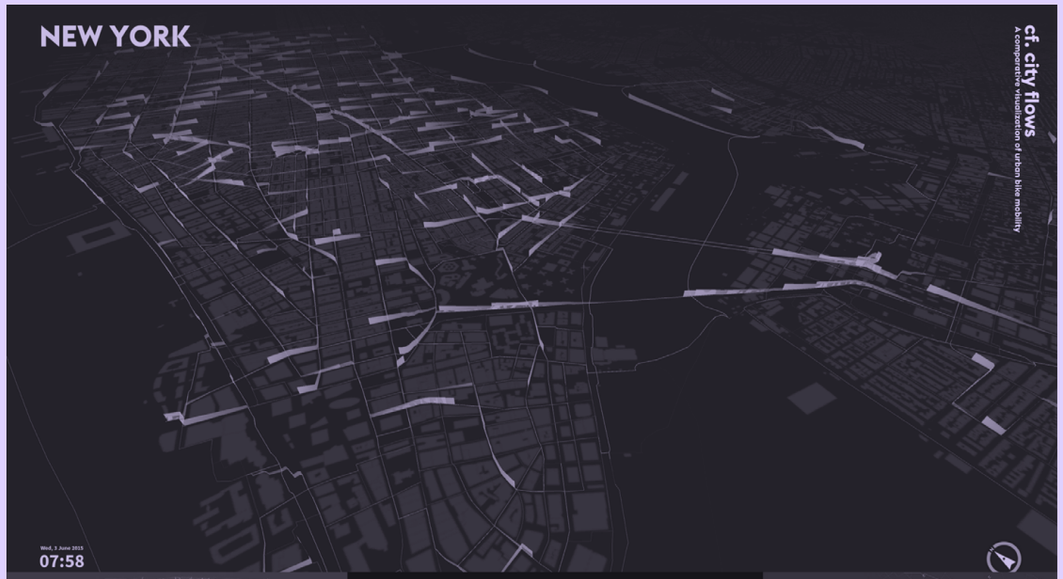


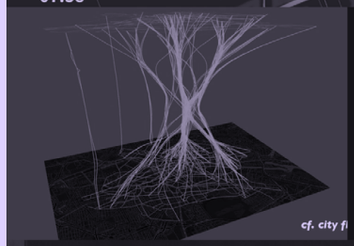


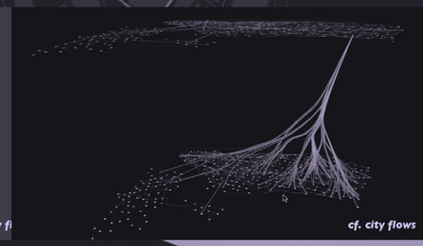


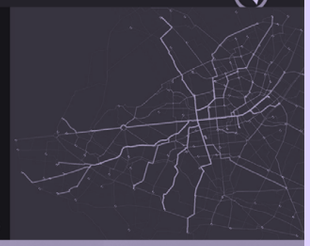












TSUBUYAKI

# #つぶやきProcessing

#つぶやきProcessing (Tsubuyaki-Processing) is a coding challenge that expresses art with Processing/p5.js code that fits in just one tweet originated by @Hau_kun san. I have been totally into it since I started in April 2020.

Creating art in the limitation gives me inspiration and joy. It is like writing Japanese Haiku by code. Here are some of my works. If you are interested, please search the hashtag and take a look at everyone's works. #つぶやきProcessing is always waiting for your challenge!

```
t=0
draw=_=>{t++||createCanvas(W=400,W)
background(0)
j=0
stroke`#つぶやきProcessing`
T=translate
R=rotate
P=push
n=noise
w=W/2
for(i=9;i--;pop(b(i*5+20)))P()+T((n(i)*W*3+t)%(W+W)-w,W)}
b=l=>l>6&&(line(0,0,0,-l),P(T(0,-l))<R(a=n(t/w,l))<P(pop(
b(l*=n(j++)+.22)))<R(-a),pop(b(l)))
```

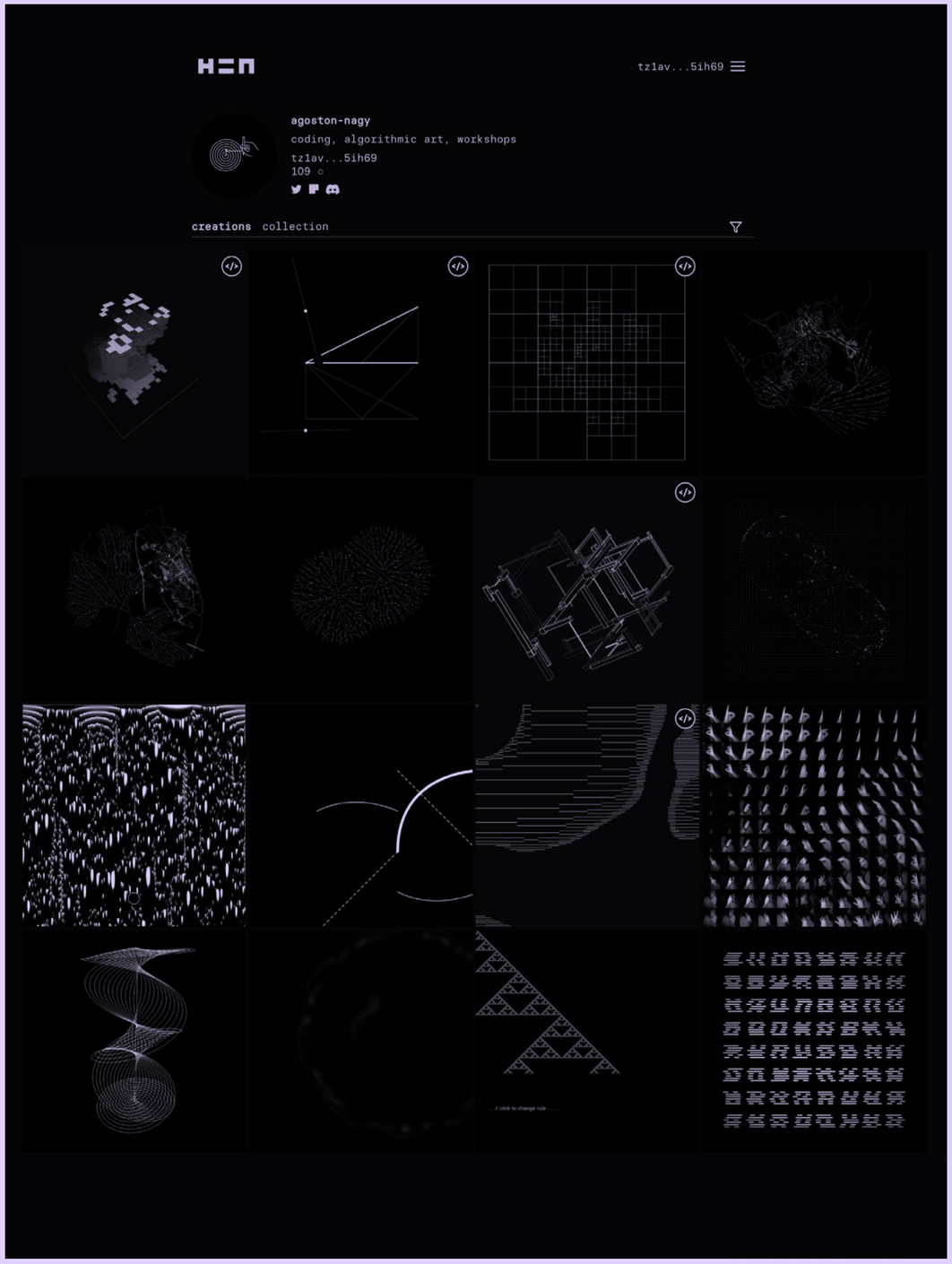
```
t=0
draw=_=>{t||createCanvas(W=600,W)
background(0)
w=W/2
noFill(stroke`#つぶやきProcessing`)
s=.05
for(x=-1;x<1;x+=s)for(y=-1;y<1;y+=s){quad(...[[x,y],[x+s,y],[x+s,y
+s],[x,y+s]].flatMap(([x,y,d=mag(x,y^2),s=t*PI/2,c=noise(cos(d+s),
sin(d+s))])=>[c*x*W+w,c*y*W+w]))}t+=.01}
```

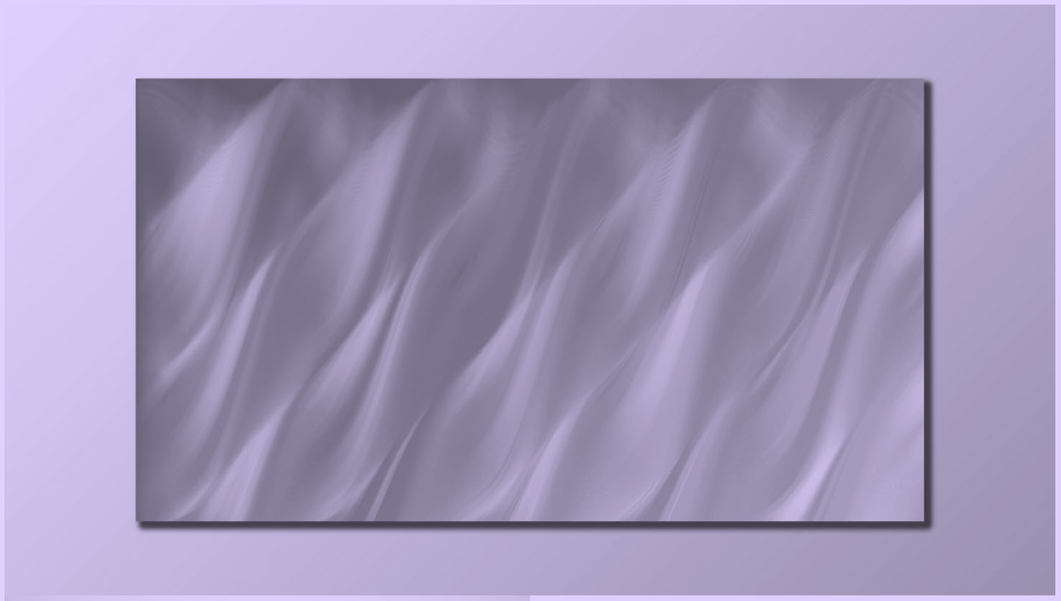
```
t=0
setup=_=>createCanvas(W=600,W)
draw=_=>{background(0)
stroke("#つぶやきProcessing")
scale(3.5)
t+=.004
for(y=29;y--;)for(x=43;x--;)for(i=15;i--;)
unchar("筋璚珧禧隠秏篏鉨銭积荏昷扎楏箷"[int(noi
se(x/99+t,y/99+t,t/3)*99%15)])&1<<i&&point
(i%3+4*x,int(i/3)+6*y)}
```

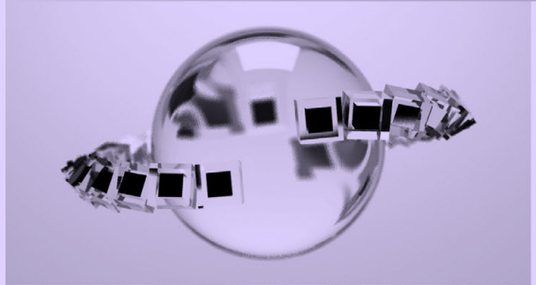
```
t=0
draw=_=>{t||createCanvas(W=600,W)
background(f=0,9)
noStroke(fill`#つぶやきProcessing #p5js`)
for(x=W;x-=120;)for(f^=1,y=W;beginShape(c=
constrain),y-=120;endShape())for(i=4;i--;)
{r=(TAU/4*i+t)*(f?1:-1)*noise(x,y,)
vertex(x+c(cos(r)*240,-60,60),y+c(sin(r)*2
40,-60,60))}t+=.02}
```

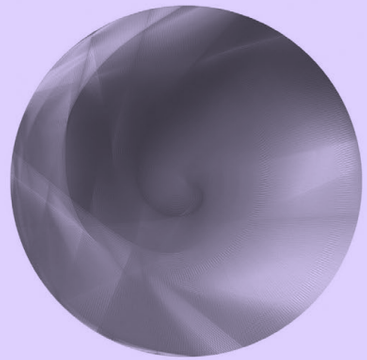
```
t=0//#つぶやきProcessing
draw=_=>{t||createCanvas(W=600,W)
background(0)
stroke(x=y=w=W/2)
circle(x,y,9)
for(i=0;i++<w;){r=9*i+9
circle(x,y,~~((i+t)/10)%10==1?15:2)
y+=v=sin(n=int(noise((t+i)/99)*19)*PI/4)*6
x+=cos(n)*6}t++}v
```

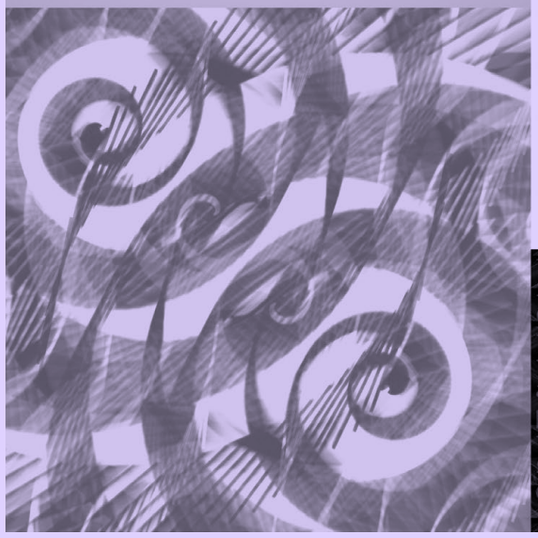
https://ngym.org/tsubuyaki-processing
2021 @nagayama


tz1av...5ih69
agoston-nagy
coding, algorithmic art, workshops
tz1av...5ih69
109
creations collection

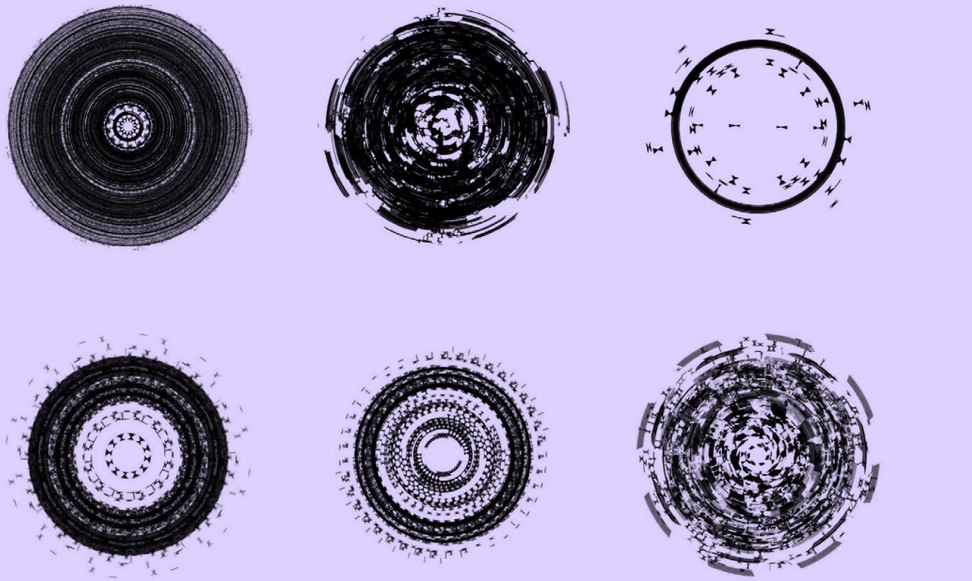


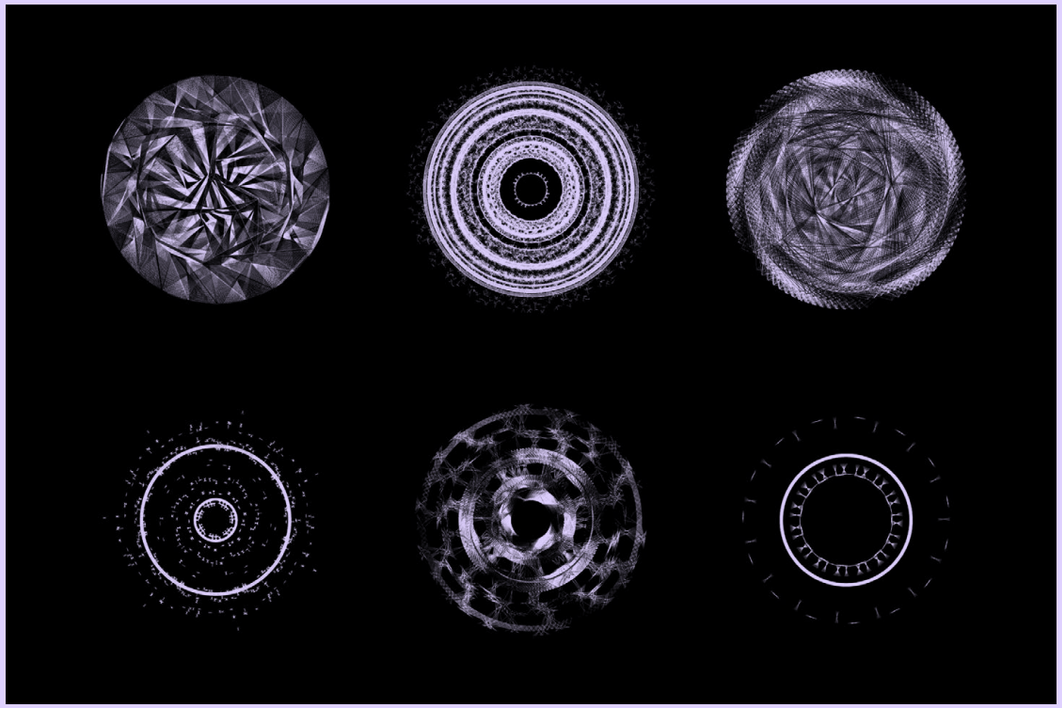




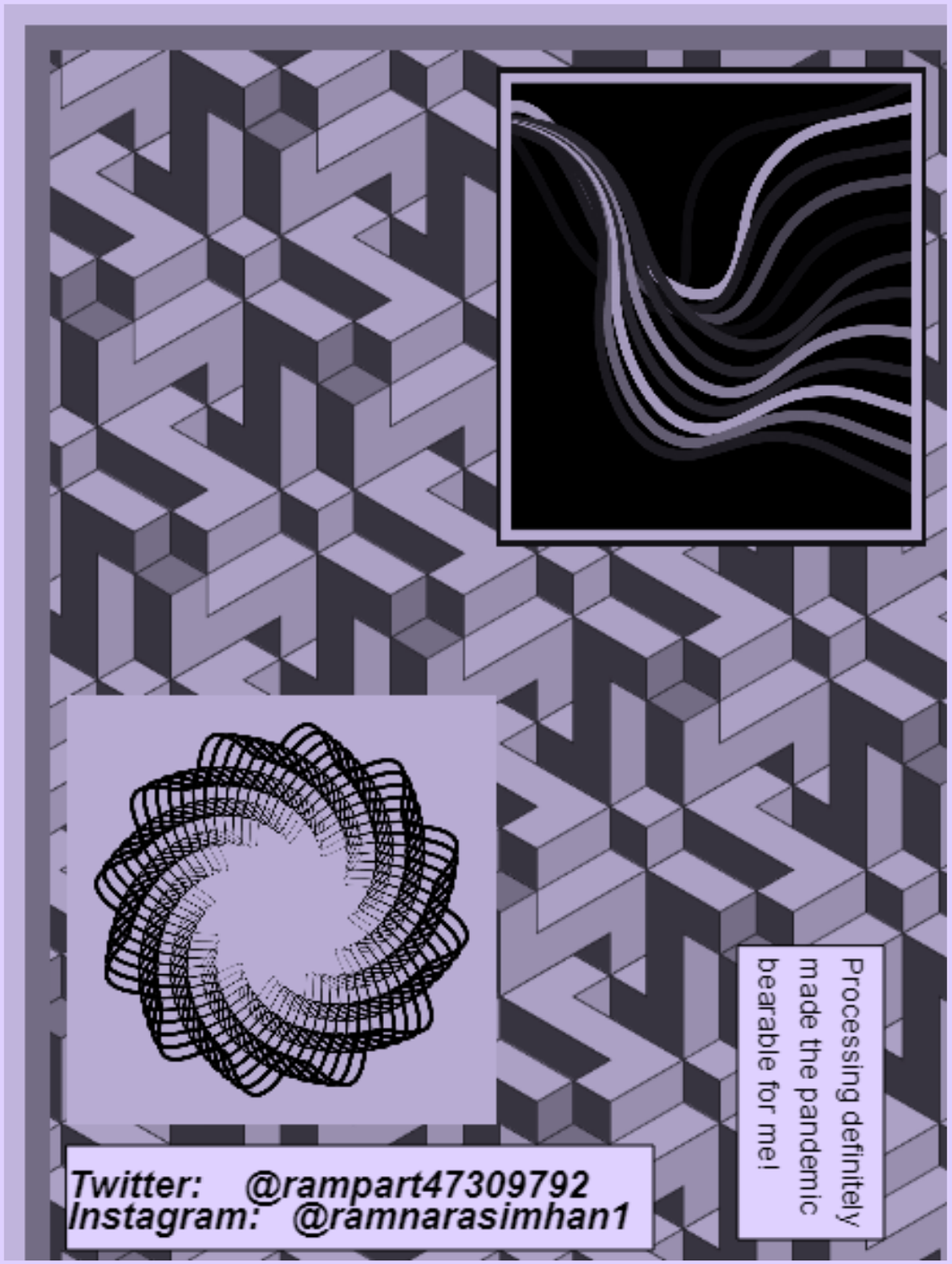



Yota Nakamura
ohayota @ohayoooota
Creative Coding Circle
ARTiS

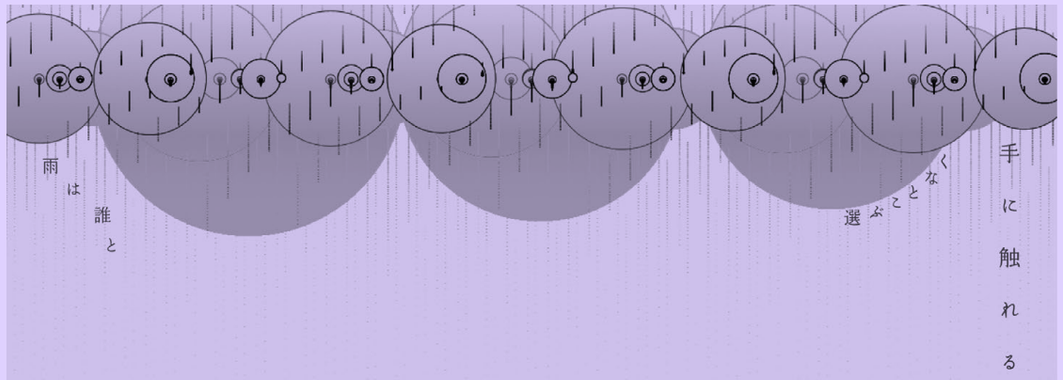







Twitter: @rampart47309792
Instagram: @ramnarasimhan1
Processing definitely made the pandemic bearable for me!



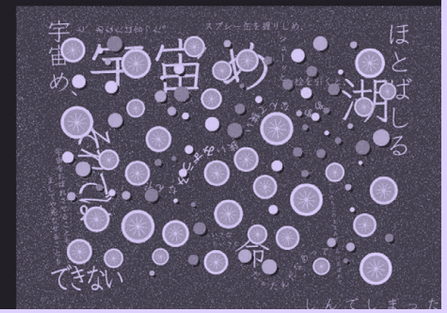


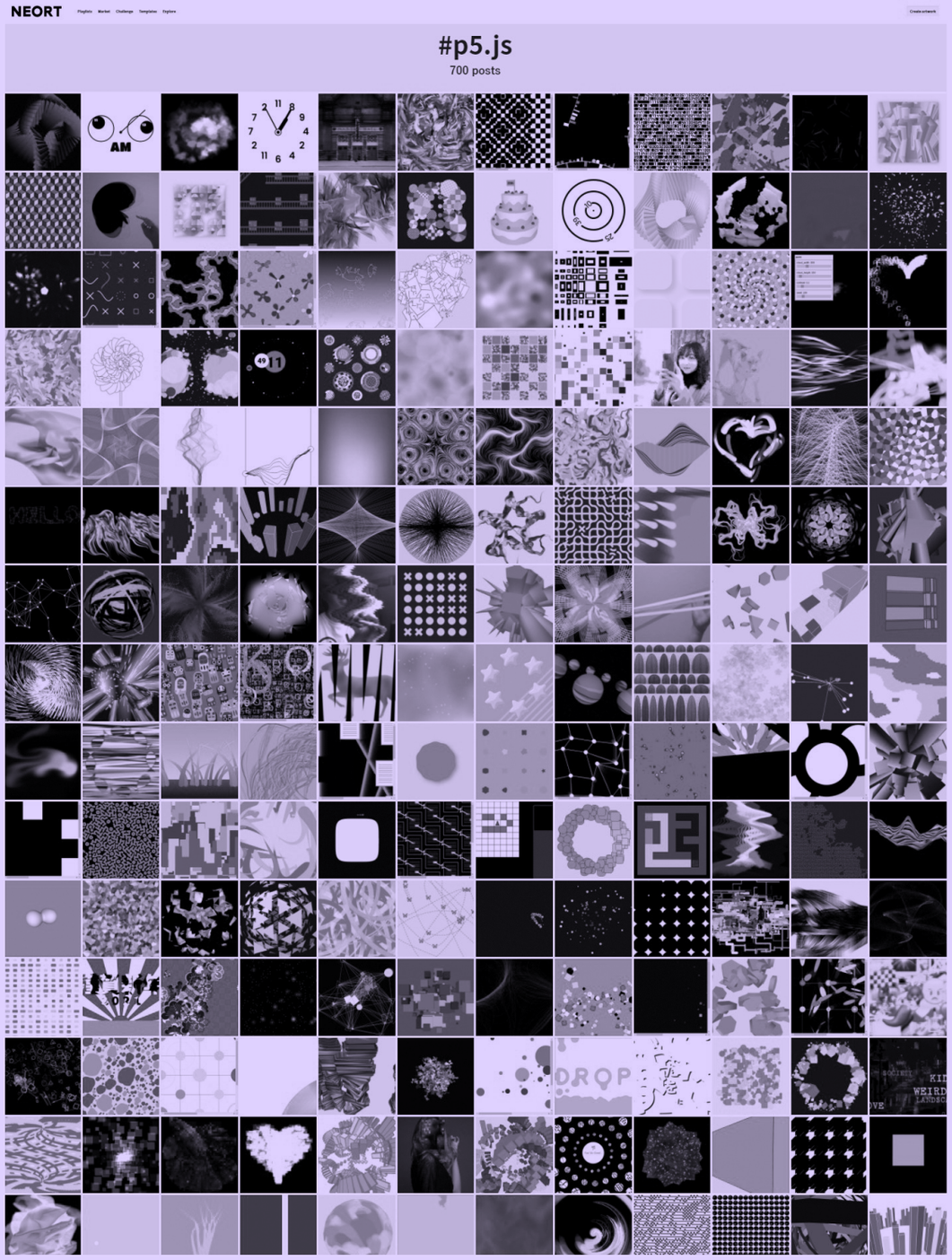

# Sculpting Words

I am **Nejio**. I'm a student at a university in Japan, and I'm working with p5js. Ever since I was a child, there is one thing that has always been close to me, and yet I have always pursued it. That is words. I have always been fascinated by words, and while I continue to create works in p5js, this desire has not changed, and I wanted to collaborate my words with p5js works. I like the meanings of words, but I believe that the words in my works have been liberated from their meanings and sublimated into figures. These works are a part of that challenge.

---

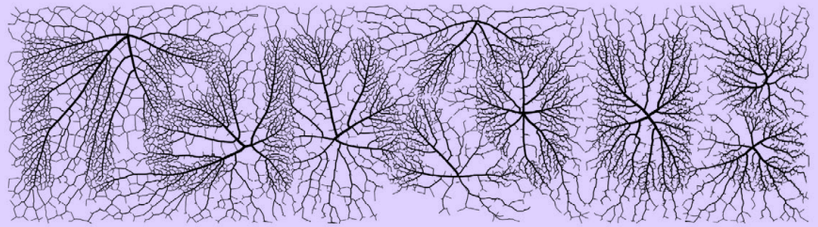
twitter @shirasu_nejio
instagram @mengyenejio



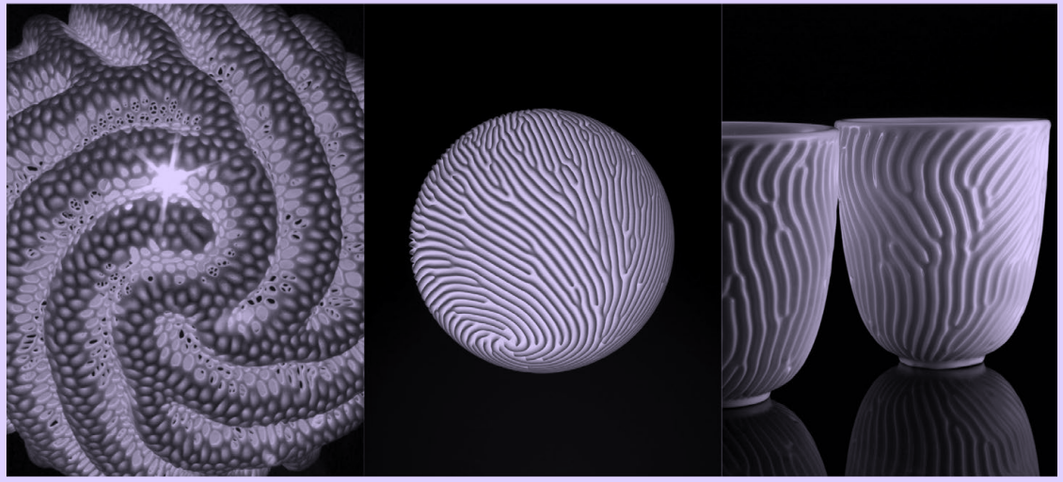





Nervous System (Jessica Rosenkrantz and Jesse Louis-Rosenberg) http://nervo.us

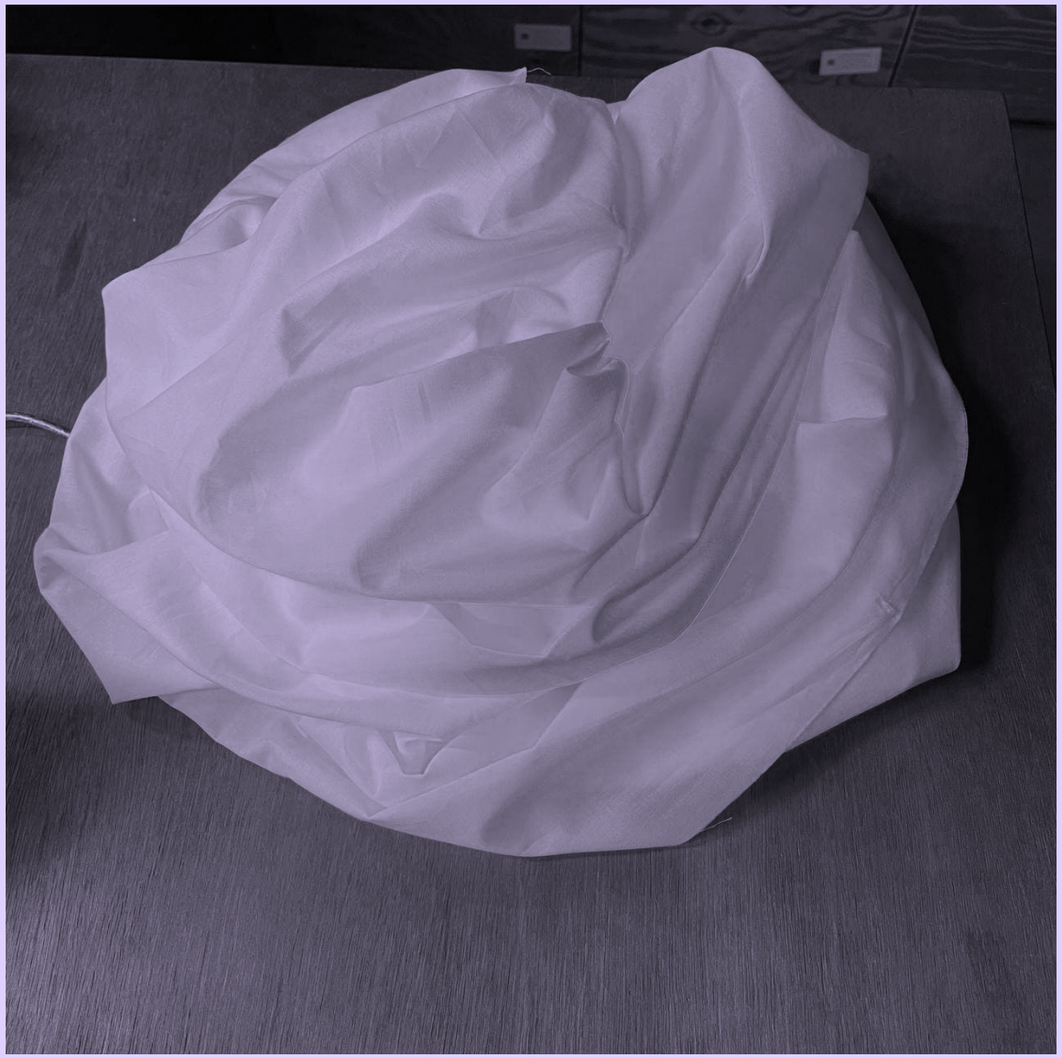


**Xylem** 2010
generative type experiment

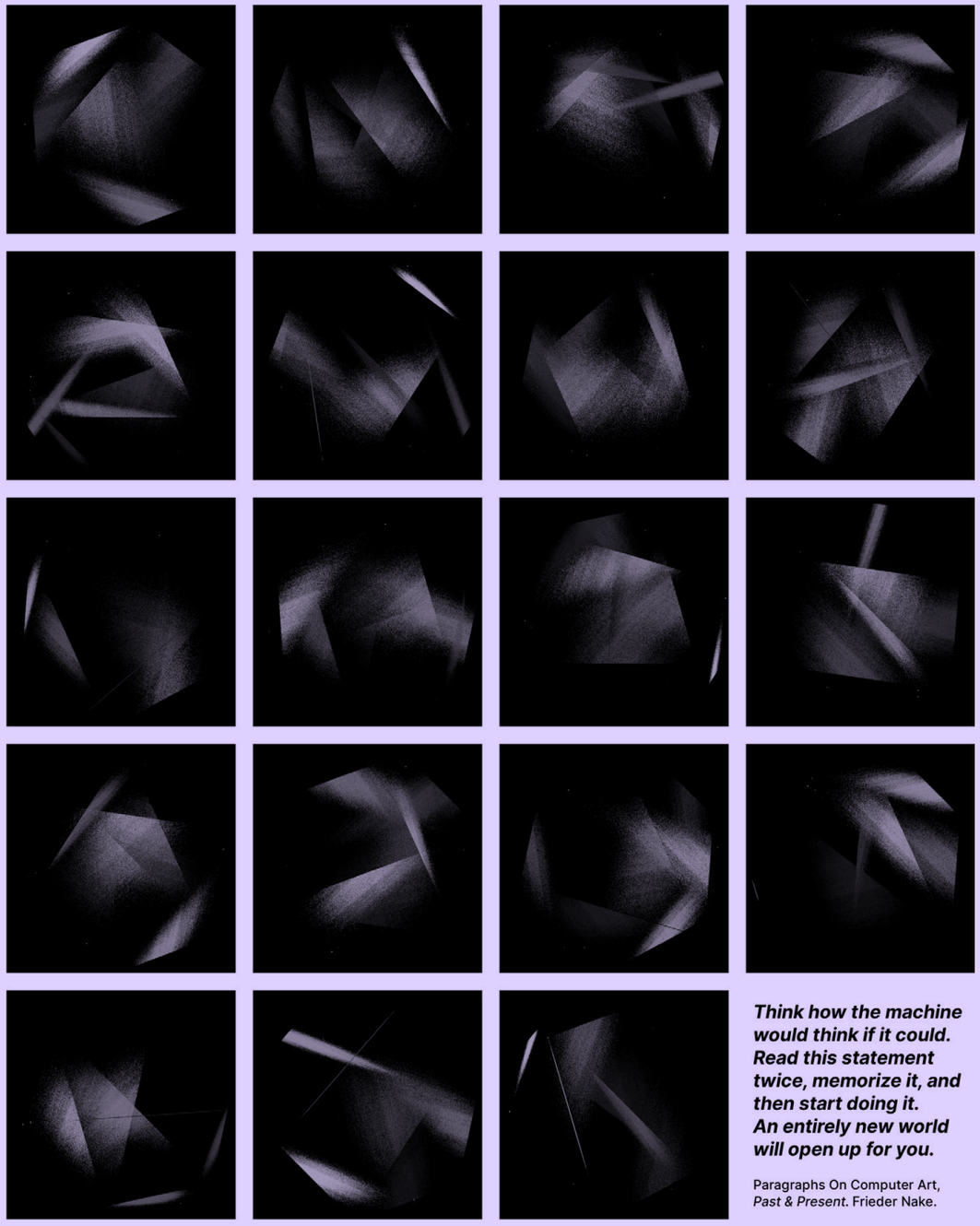


**Reaction** 2010
Seed and Reaction Lamps (3D-printed nylon), Reaction Cup (slipcast porcelain)

**Cell Cycle** 2009
Cell Cycle online customizer, Wave and Interstice bangles (3D-printed nylon), Rings (sterling silver)







**Aequilibrium trianguli**
Nicolas Lebrun

***Think how the machine would think if it could. Read this statement twice, memorize it, and then start doing it. An entirely new world will open up for you.***

Paragraphs On Computer Art, *Past & Present*. Frieder Nake.

HTTPS://TWITTER.COM/NICOLASBAEZ

CON 383

698

p5* File Edit Sketch Help English Hello, nicolasbaez!

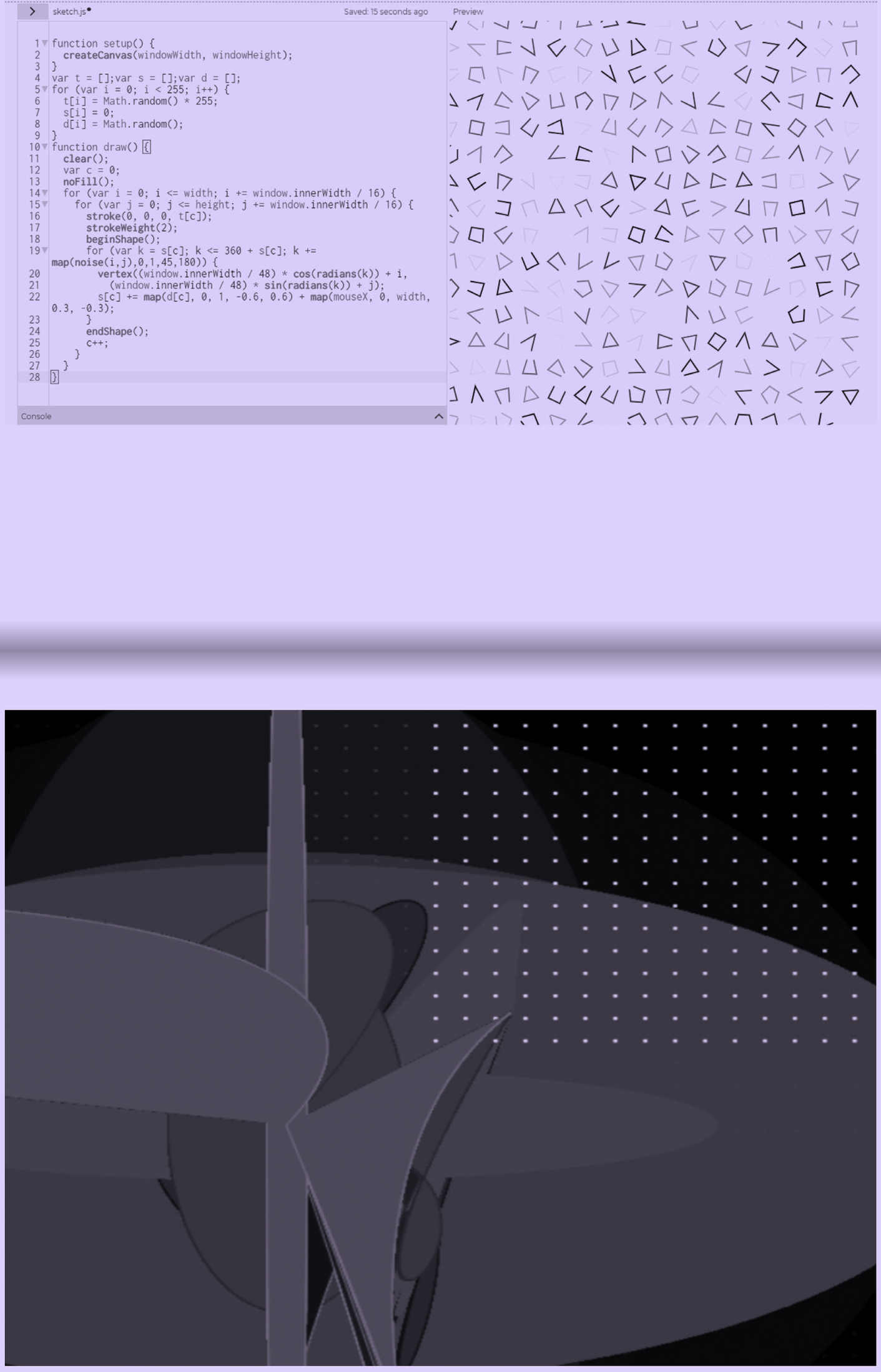
Auto-refresh Sedate mandarin by nicolasbaez

sketch.js Saved: 15 seconds ago

```
function setup() {
  createCanvas(windowWidth, windowHeight);
}
var t = [];var s = [];var d = [];
for (var i = 0; i < 255; i++) {
  t[i] = Math.random() * 255;
  s[i] = 0;
  d[i] = Math.random();
}
function draw() {
  clear();
  var c = 0;
  noFill();
  for (var i = 0; i <= width; i += window.innerWidth / 16) {
    for (var j = 0; j <= height; j += window.innerWidth / 16) {
      stroke(0, 0, 0, t[c]);
      strokeWeight(2);
      beginShape();
      for (var k = s[c]; k <= 360 + s[c]; k +=
map(noise(i,j),0,1,45,180)) {
        vertex((window.innerWidth / 48) * cos(radians(k)) + i,
          (window.innerWidth / 48) * sin(radians(k)) + j);
        s[c] += map(d[c], 0, 1, -0.6, 0.6) + map(mouseX, 0, width,
0.3, -0.3);
      }
      endShape();
      c++;
    }
  }
}
```

Preview

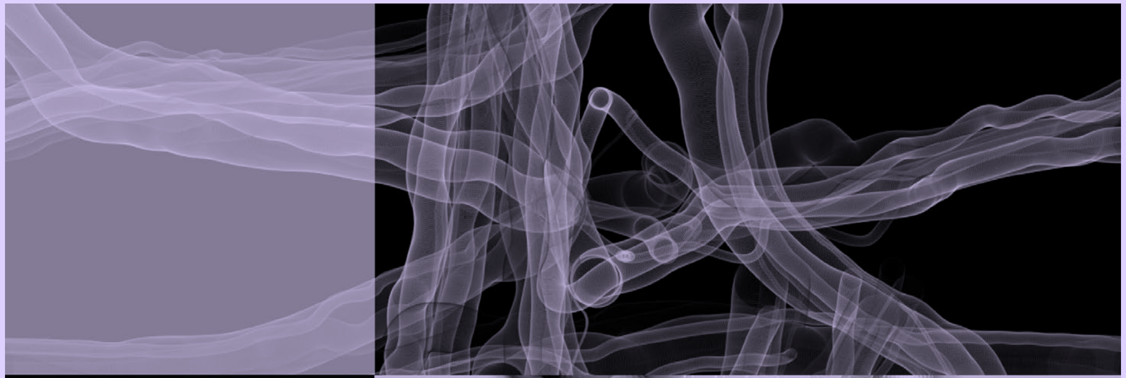
Console



HTTPS://WWW.INSTAGRAM.COM/NO.FILL_/

CON 384

699



**Widianto Nugroho**

**Responsive Painting Series**
Multiple prints from a series of generative artworks. Each of which can respond to desktop mouse, audio from surrounding noise, and motion with standalone Kinect sensor.

2009 - 2011. Variable dimension.
Built with Processing.

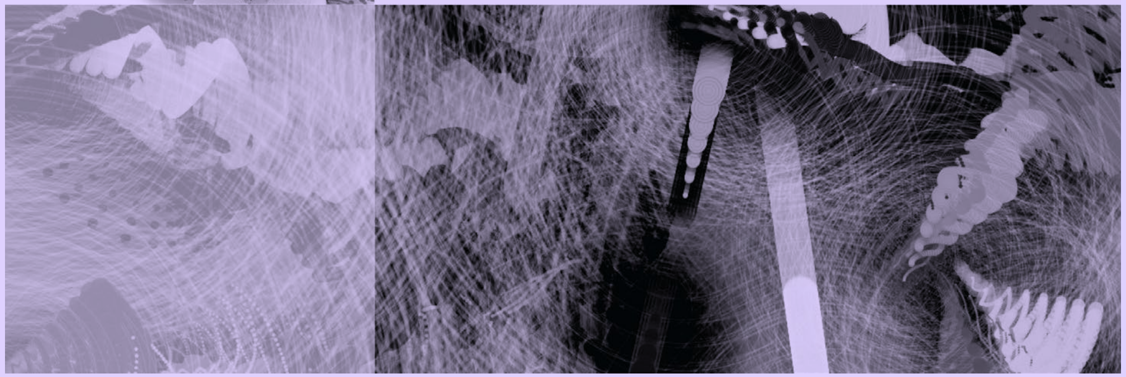
I first learned about the idea of making art by writing computer program in early 2000s when I browsed and stumbled across the Aesthetic and Computation Group (ACG) at the MIT Media Lab led by Prof. John Maeda. The group created Design By Number (DBN), then, among others, continued with Proce55ing led by Ben Fry and Casey Reas.

As an art school graduate in Indonesia I was interested to learn to code with Proce55ing which was still in its alpha version at that time.

Then I continued to explore and create several works, teach in several workshops, and exhibit works made by Processing in Bandung and Jakarta until around 2011.

I am currently on hiatus from creating art with Processing, or other related software developed with similar ideas. I really hope to be back in the near future :-)

**Widianto Nugroho**
Bandung, Indonesia



```
import com.hamoid.*;

VideoExport videoExport;



// setting up the clicking events
int Objects = 15; // number of Pac
int ClickCounter = 0;
color NotFill;
Pacman[] Pacs; //making an array o
color [] PacFill = {#F20A0A,#12FFC
float TransZ = 0; // translation


void setup() {
  size(1920, 1070,P3D);
```

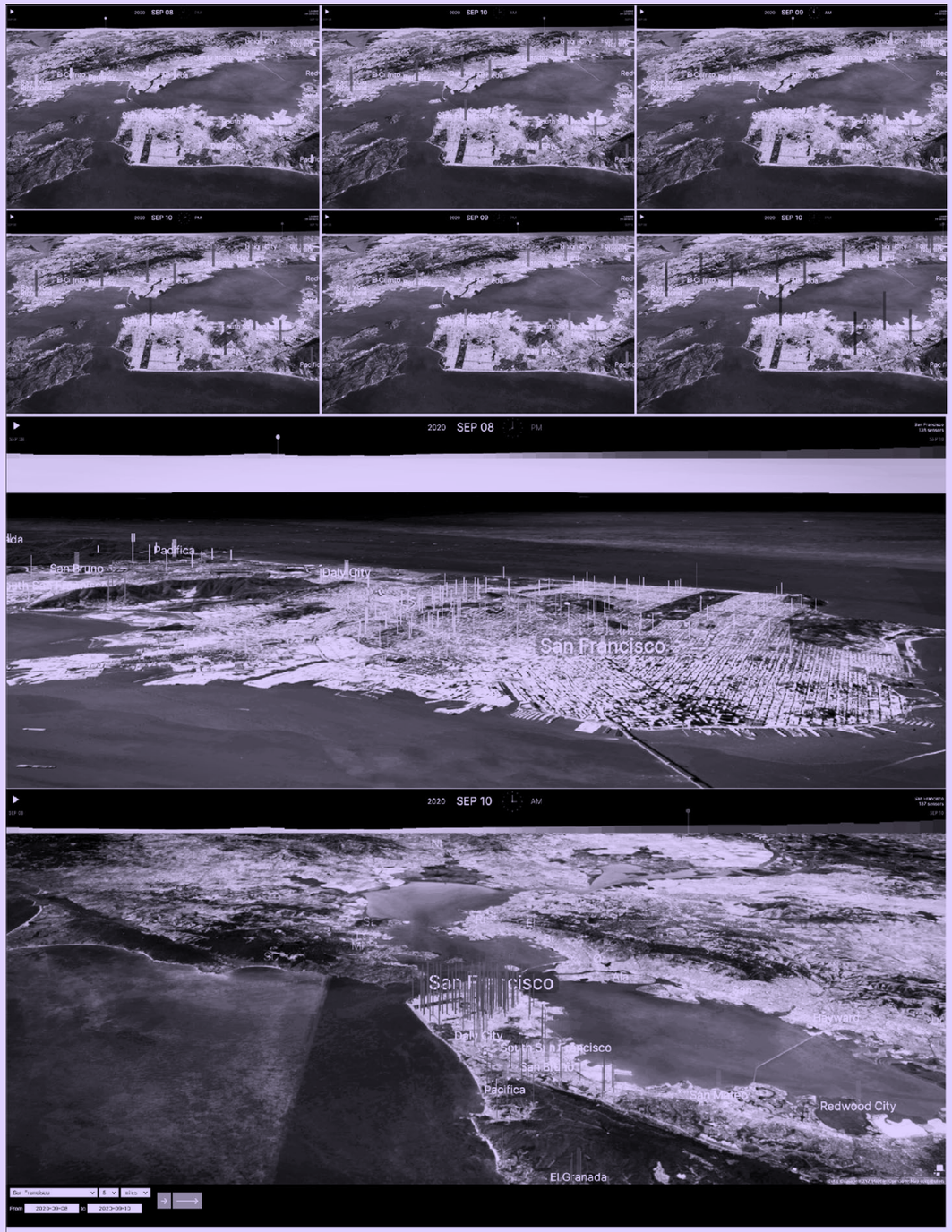
**Code is Poetry and Poetry is code, Processing as a canvas as my Creativity Grows**




# Processing 20 years!

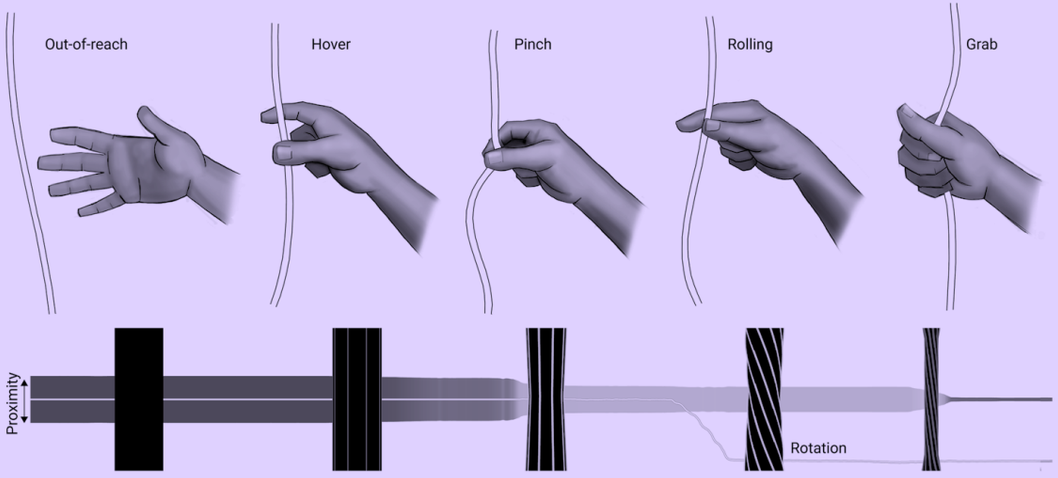
```
import krister.Ess;

AudioChannel myChannel;
SineWave myWave;

void setup() {
  size(256, 200);

  // Start up Ess
  Ess.start(this);

  // Create a new AudioChannel
  myChannel = new AudioChannel();

  // Set the channel size to 5 seconds
  myChannel.initChannel(myChannel.frames(5000));

  // Generate a 3 second sine wave
  myWave = new SineWave(480, 0.1);
  myWave.generate(myChannel, 0, myChannel.frames(9000));

  // Play
  myChannel.play();
}

void draw() {
}

// We are done, clean up Ess
public void stop() {
  Ess.stop();
  super.stop();
}
```






BAY AREA AIR QUALITY: INTERACTIVE TIME SERIES VISUALIZATIONS OF AIR QUALITY SENSOR DATA (p5.js + Procedural.gl) ALEX OLWAL www.olwal.com

I/O Braid is an interactive textile cord with embedded sensing and visuals. It senses user's proximity, touch, and twist, while embedded fiber optics enable spiraling light feedback.

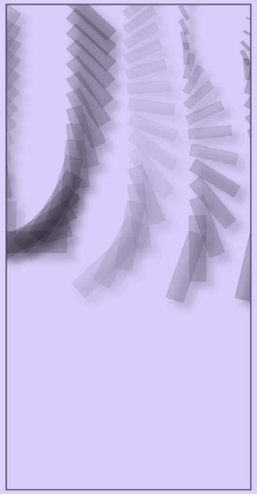



Alex Olwal  Jon Moeller  Greg Priest-Dorman  Thad Starner  Ben Carroll  olwal.com/iobraid

We thank the Google ATAP Jacquard team for our collaboration, especially Shiho Fukuhara, Munehiko Sato, and Ivan Poupyrev. We thank the Google Wearables team, and Kenneth Albanowski, in particular, for engineering support. We thank Mark Zarich for illustrations, Bryan Allen for videography, and Carolyn Priest-Dorman for advice on textile techniques.



*Dear Processing,*



I created this image using Processing as a motif for Tanabata, a festival in Sendai where I was born and raised.

My first encounter with Processing was at the university library.
Among the shelves of art and design books that I often browsed, I found a book about expression through code.

I was studying mechanical engineering at the time and was tired of the math and physics lectures,
so I spent most of my time listening to and creating music.

I thought of mathematics and engineering as completely different from creative activities.

Processing changed my mind.

I was so excited when I realized that the field I was studying was connected to creativity, and it changed the rest of my life.
Now, as an engineer, I can work with many different artists.

I think that Processing has taught me that creativity exists in any field.

Congratulations on your 20th anniversary !

この画像は、私が生まれ育った仙台という街のお祭りである七夕をモチーフにProcessingで書いたものです。

Processingとの出会いは大学の図書館でした。
よく眺めていたアートやデザインの棚の中から、コードによる表現について書かれた本を見つけたのです。

当時機械工学を学んでいた私は、数学や物理学の講義に疲れていて、もっぱら音楽を聴いたり創ることに没頭していました。
数学や工学と創作活動は全く別のものと捉えていたのです。

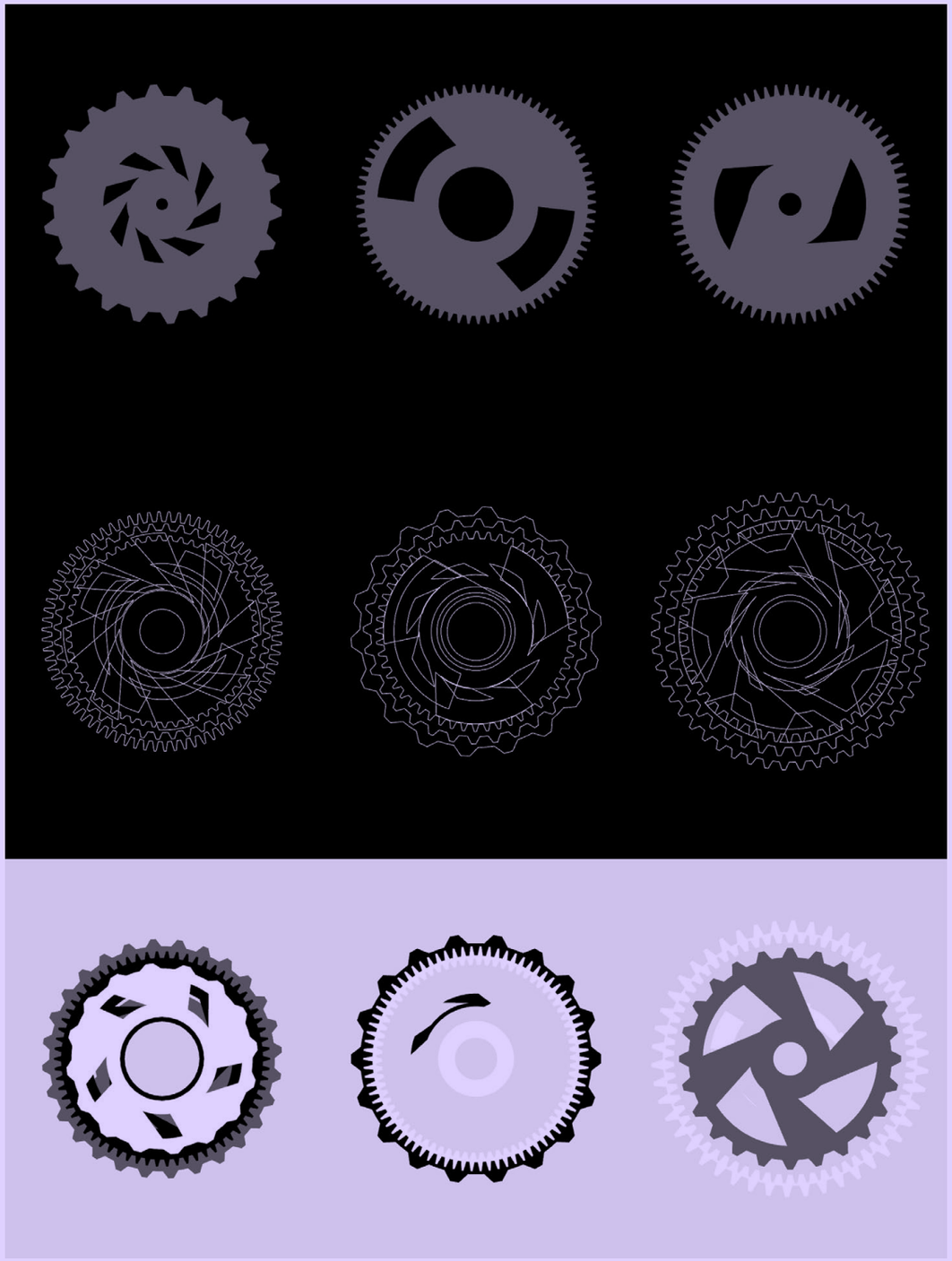
その考えを大きく変えてくれたのがProcessingでした。

自分が学んでいる分野が創造性に結びついていると気づかされた時はとても興奮して、その後の人生が変わったほどです。
今ではエンジニアとして、様々なアーティストと仕事ができるようになりました。

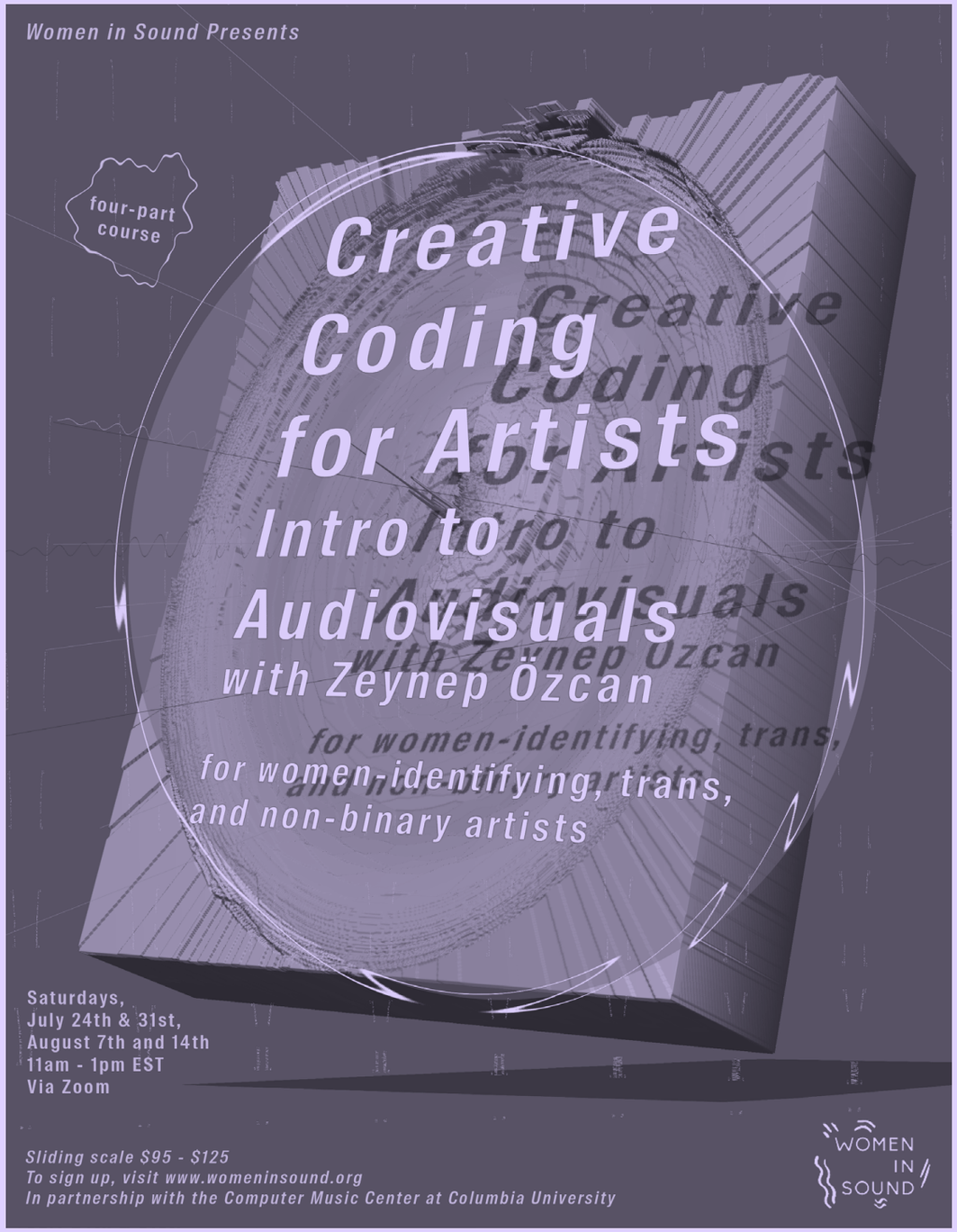
どんな分野であっても創造性は存在するのだと、Processingは教えてくれたのだと思います。

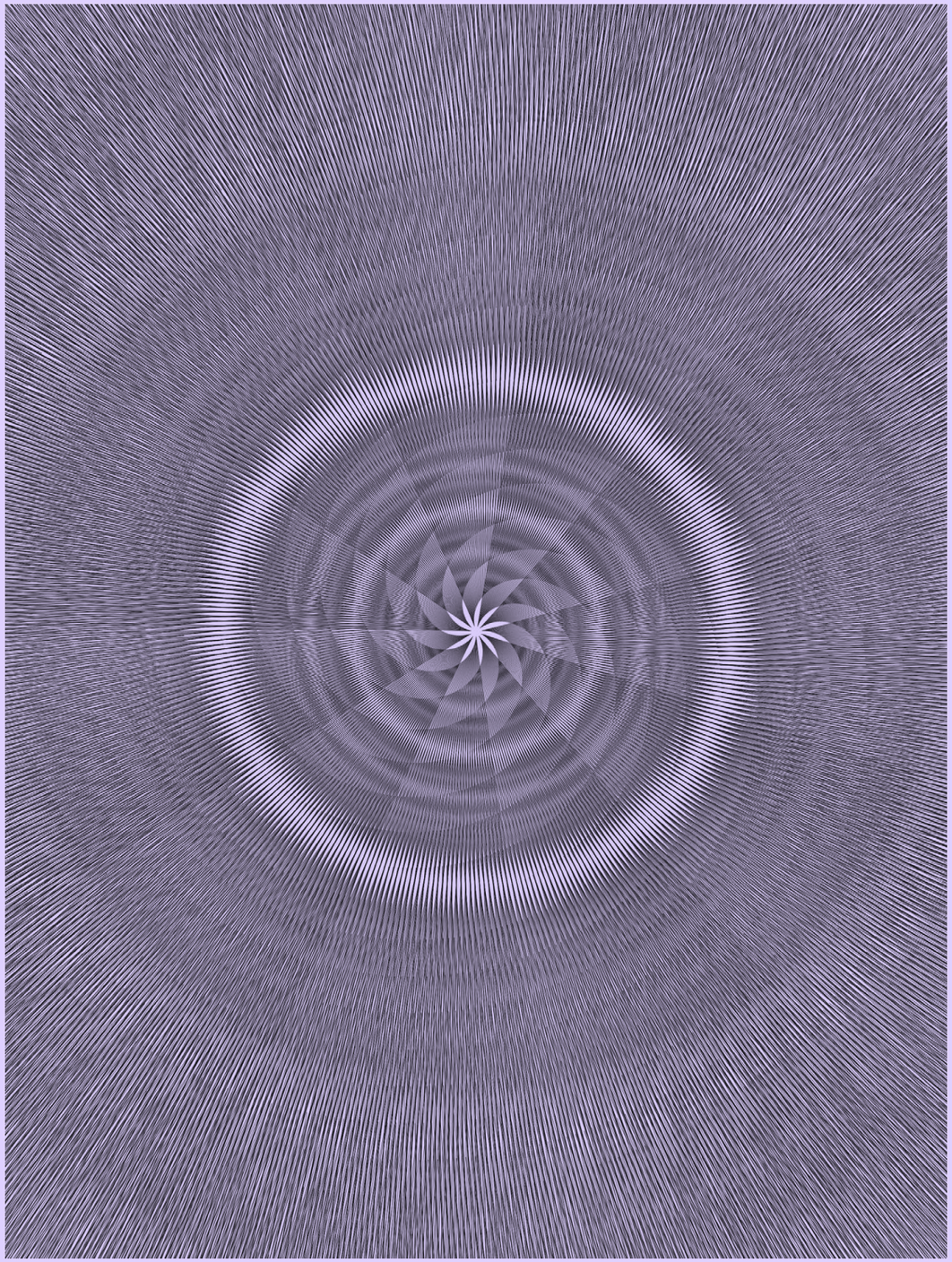
20周年おめでとうございます！

Takafumi Oyama, from Japan






Women in Sound Presents
four-part course
Creative Coding for Artists
Intro to Audiovisuals
with Zeynep Özcan
for women-identifying, trans, and non-binary artists
Saturdays,
July 24th & 31st,
August 7th and 14th
11am - 1pm EST
Via Zoom
Sliding scale $95 - $125
To sign up, visit www.womeninsound.org
In partnership with the Computer Music Center at Columbia University
WOMEN IN SOUND

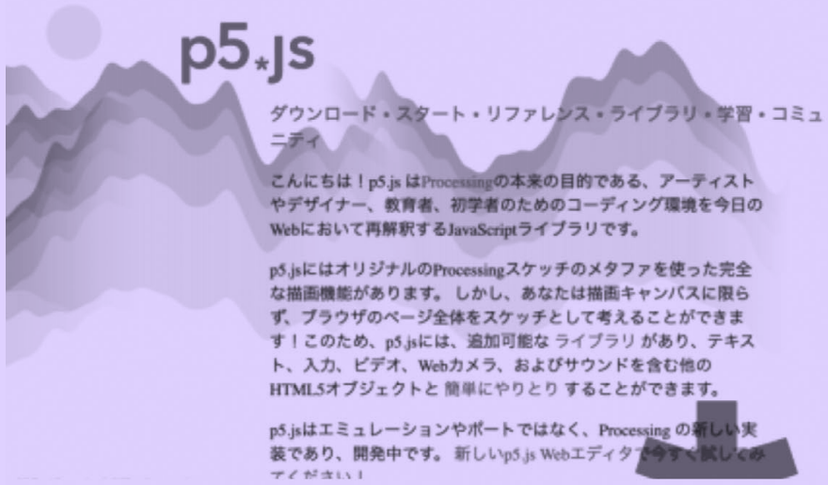




# Japanese translation project for p5.js website

The mission of our team is to translate the p5.js website from English to Japanese and provide j5.js resources in Japanese through Japansese version of p5.js website.
I am Katsuya Endoh who leads this team. A reason why we have tackled to translate p5.js website and contribute to j5.js community from 2017, we had learned so much through Processing.
When I contact with p5.js community that I had translated the website into Japanese through GitHub, they graciously agreed to let us get the domain and host it. However, I found that it difficult to manage the website by myself. So, I have been working on the translation with students from Meisei University since September 2021. We've just worked together, however we are good team.
I have learned many things through Processing, such as coding, art, and how to think about to OSS. That's why I'd like to contribute to the community through this work. I hope that the students who are work with me will learn a lot from this project as well.

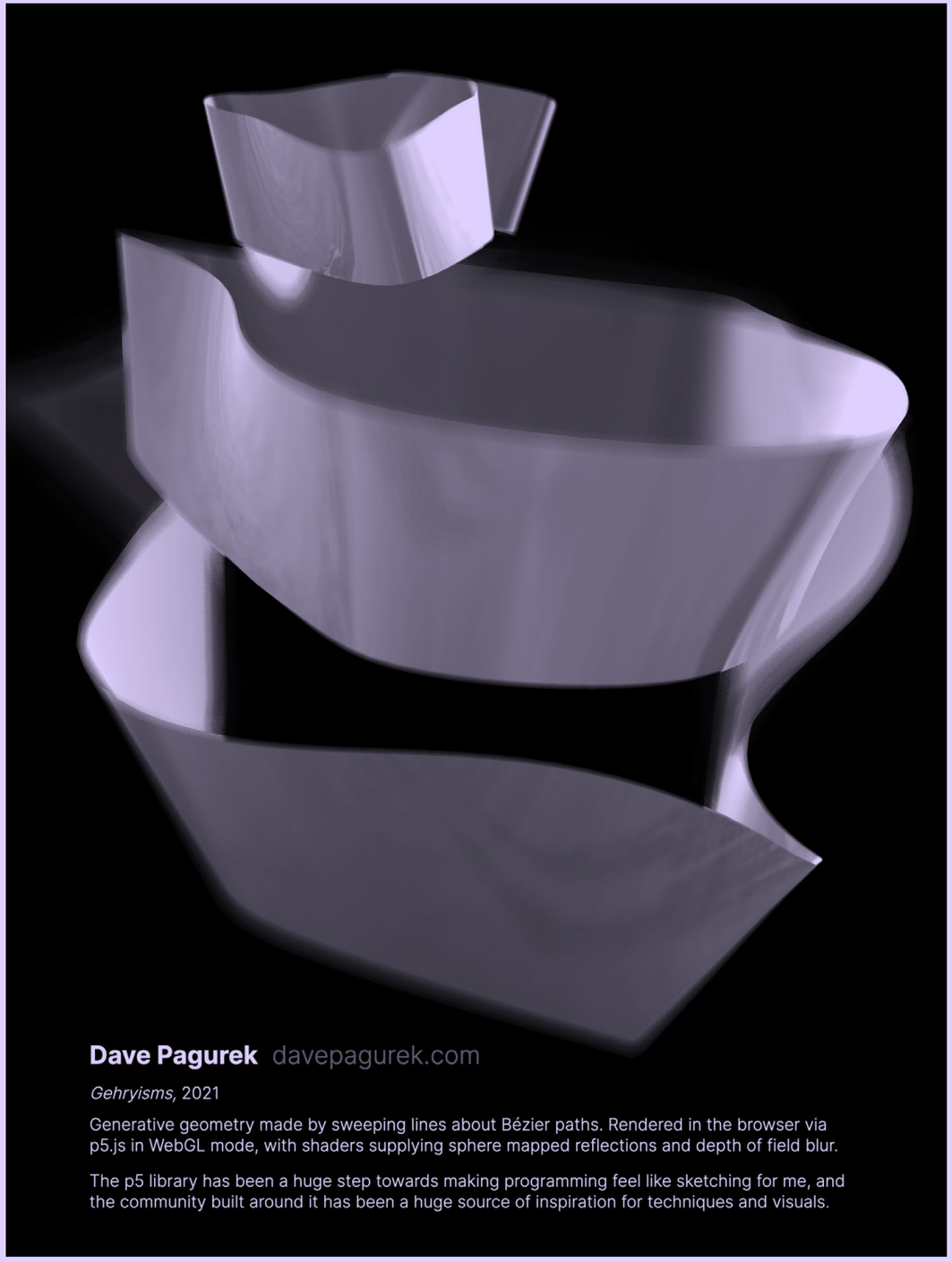
Project lead
Katsuya Endoh




**Japanese Translation Team**
Katsuya Endoh, Kaito Shimizu, Katsumi Masuda, Madoka Abe,
Miu Kanematsu, Yuma Okamura, Yuta Tamamura,
Ben Watanabe, Funa Maruyama, Koki Honda, Yu Nitta,
Makoto Yamamoto, Yuki Nakamura,
Kota Kikuchi, Takuya Taketomi and Toshitaka Amaoka

https://p5js.jp/



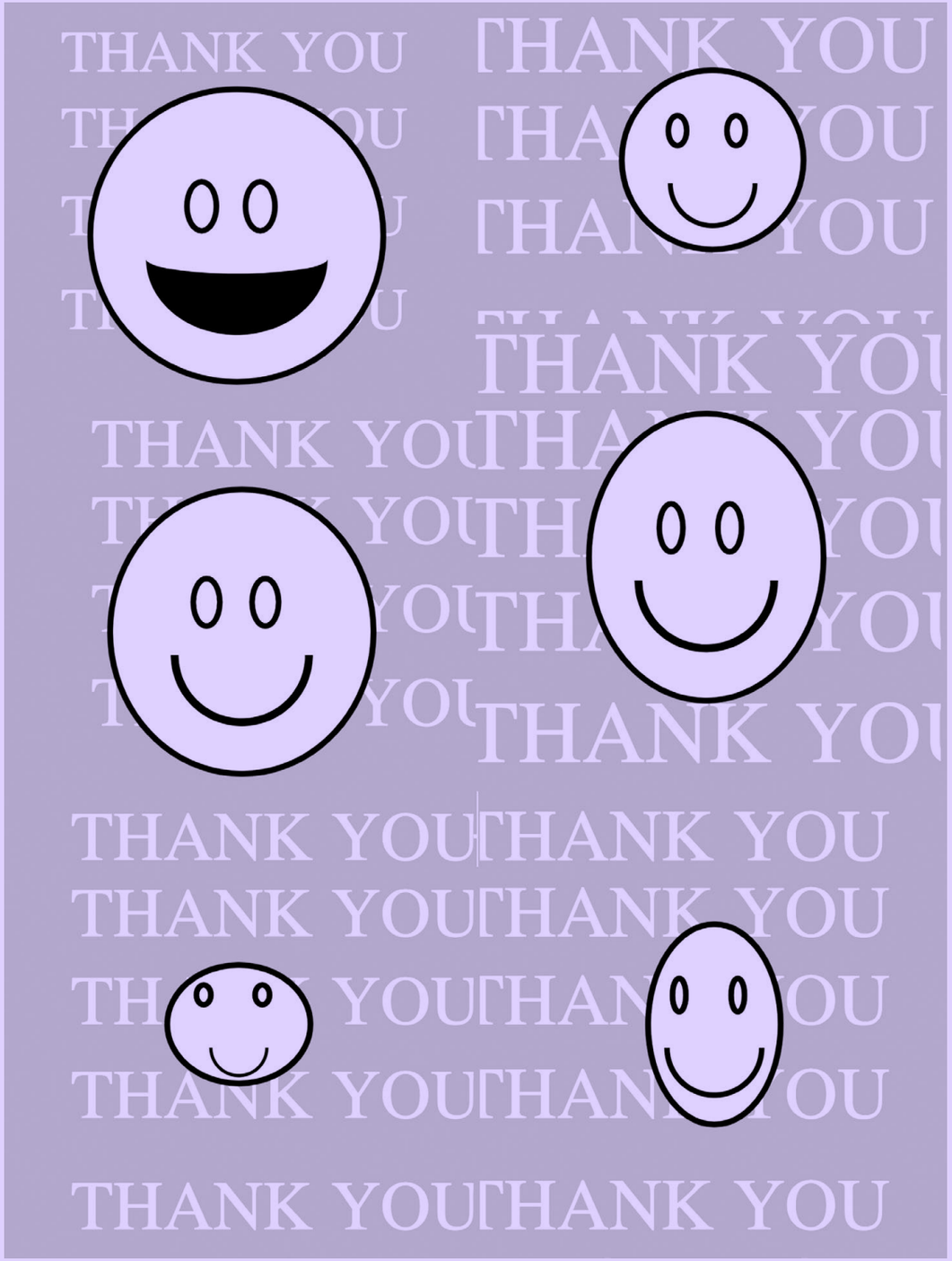


**Dave Pagurek** davepagurek.com

*Gehryisms,* 2021

Generative geometry made by sweeping lines about Bézier paths. Rendered in the browser via p5.js in WebGL mode, with shaders supplying sphere mapped reflections and depth of field blur.

The p5 library has been a huge step towards making programming feel like sketching for me, and the community built around it has been a huge source of inspiration for techniques and visuals.



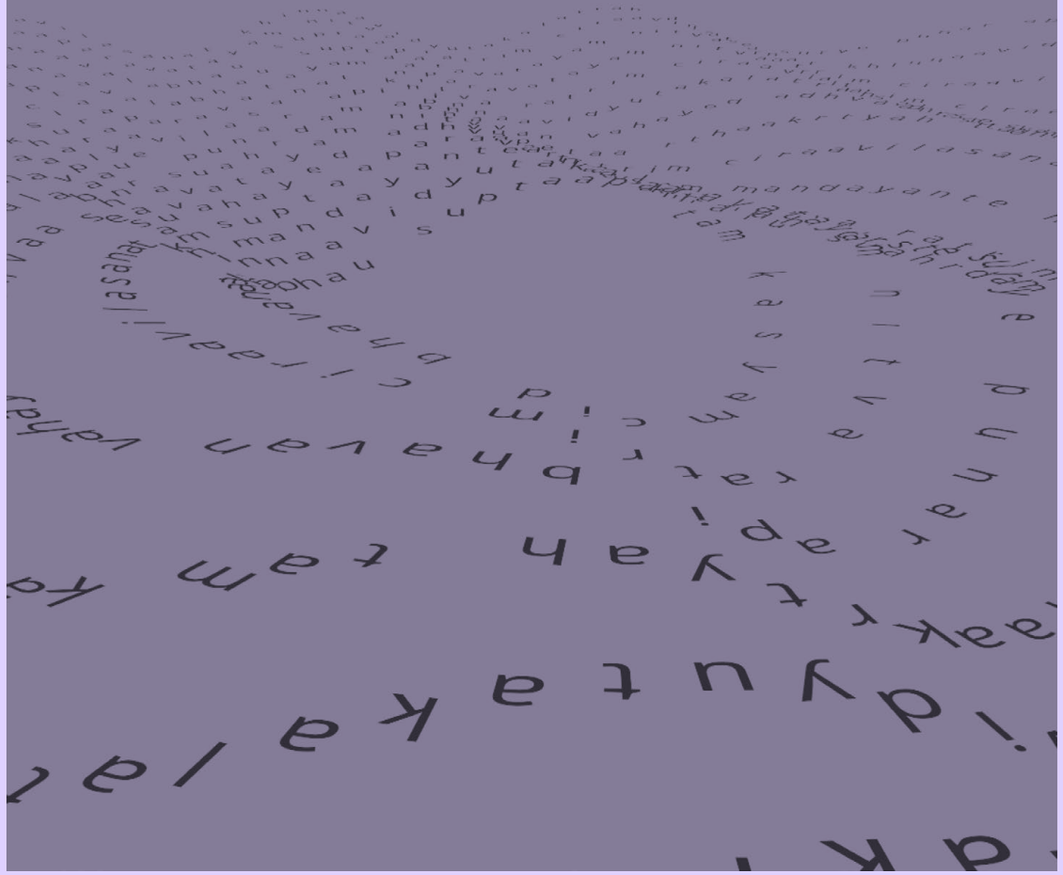



# Asking a Shadow to Dance

**Priti Pandurangan**
meghadutam.priti.is
Instagram : priti.pg

This project envisions Kalidāsā's venerable poetry — Meghadūtam — as interactive, three-dimensional typographic forms, moving through space. Type is viewed through the lens of computation to weave form, motion and interaction to take readers through the poem's emotional journey.

If poetry is a mere shadow of emotions felt, by asking a shadow to dance, we urge to bring words to life, to hold a stronger presence, a deeper voice.



Conversation 1 | Commits 1 | Checks 0 | Files changed 2 | +239 −239

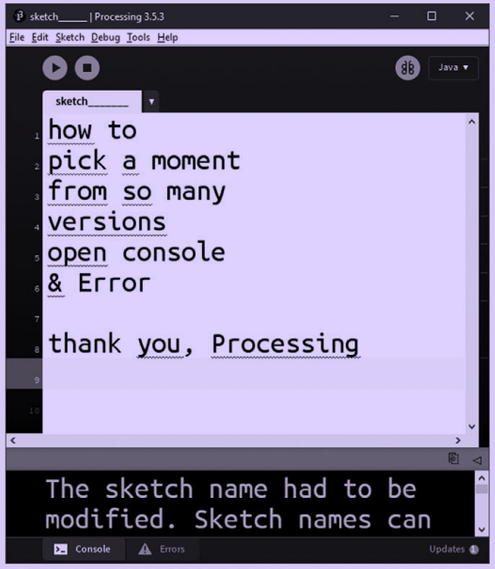
Changes from all commits | File filter | Conversations | Jump to

**Greek translation**

Arty2 committed on Aug 1, 2014 commit f3f2f8ac6fddda88d5e1642d7662f0c5973ad712

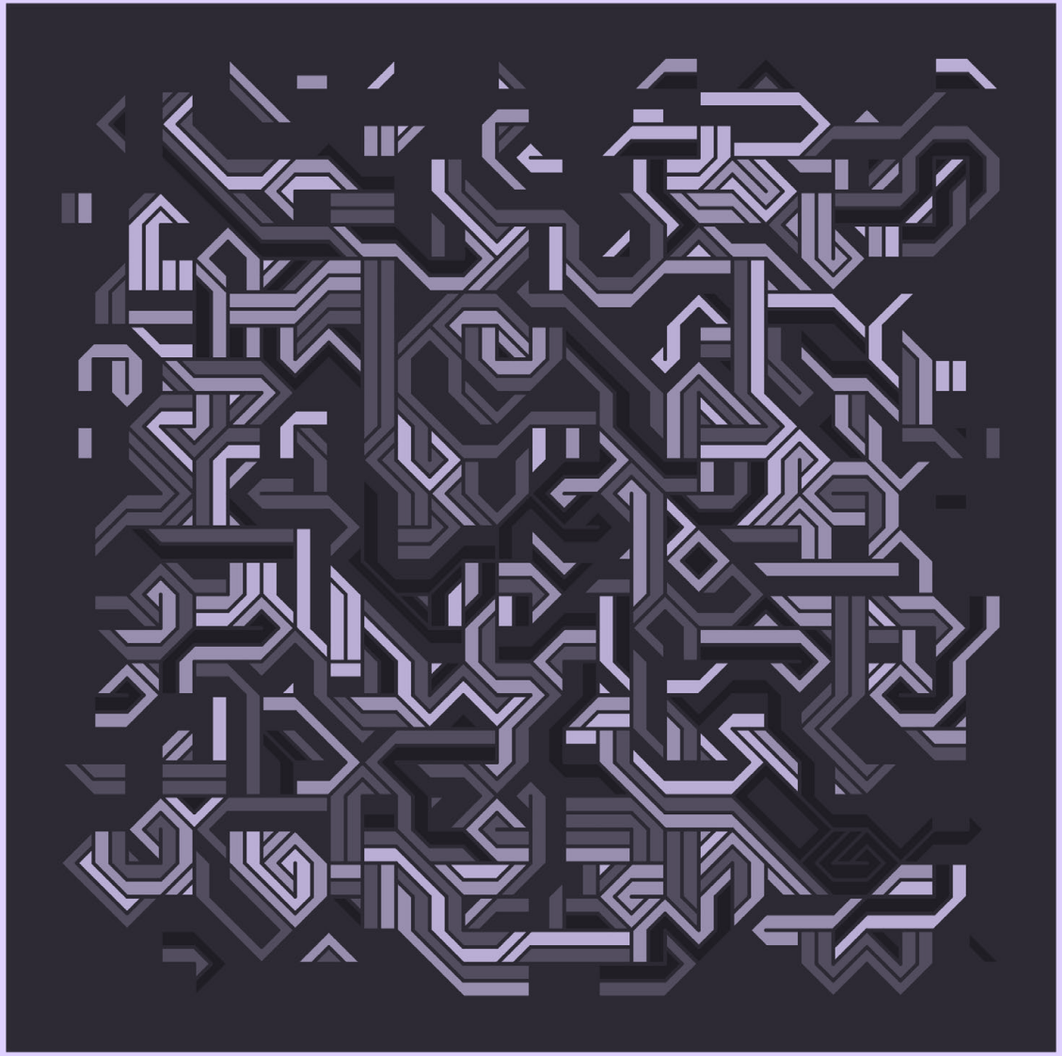
239 app/src/processing/app/languages/ PDE_el.properties

```
@@ -0,0 +1,239 @@
+
+
+ # --------------------------------------
+ # Language: Greek (Ελληνικά) (el)
+ # --------------------------------------
+
+
+ # --------------------------------------
+ # Menu
+
+ # | File | Edit | Sketch | Library | Tools | Help |
+ # | File |
+ menu.file = Αρχείο
+ menu.file.new = Νέο
+ menu.file.open = Άνοιγμα...
+ menu.file.sketchbook = Σχεδιοθήκη
+ menu.file.recent = Πρόσφατα
+ menu.file.examples = Παραδείγματα...
+ menu.file.close = Κλείσιμο
+ menu.file.save = Αποθήκευση
+ menu.file.save_as = Αποθήκευση ως...
+ menu.file.export_application = Εξαγωγή Εφαρμογής...
+ menu.file.page_setup = Διαμόρφωση σελίδας
+ menu.file.print = Εκτύπωση...
+ menu.file.preferences = Προτιμήσεις...
+ menu.file.quit = Έξοδος
+
+ # | File | Edit | Sketch | Library | Tools | Help |
+ #        | Edit |
+ menu.edit = Επεξεργασία
+ menu.edit.undo = Αναίρεση
+ menu.edit.redo = Επανάληψη
+ menu.edit.cut = Αποκοπή
+ menu.edit.copy = Αντιγραφή
+ menu.edit.copy_as_html = Αντιγραφή ως HTML
+ menu.edit.paste = Επικόλληση
+ menu.edit.select_all = Επιλογή όλων
+ menu.edit.auto_format = Αυτόματη διαμόρφωση
+ menu.edit.comment_uncomment = Σχολιασμός/Αποσχολιασμός
+ menu.edit.increase_indent = \u2192 Αύξηση εσοχής
+ menu.edit.decrease_indent = \u2190 Μείωση εσοχής
+ menu.edit.find = Αναζήτηση...
+ menu.edit.find_next = Αναζήτηση επόμενου
+ menu.edit.find_previous = Αναζήτηση προηγούμενου
+ menu.edit.use_selection_for_find = Χρήση επιλογής για Αναζήτηση
+
+ # | File | Edit | Sketch | Library | Tools | Help |
+ #              | Sketch |
+ menu.sketch = Σχέδιο
+ menu.sketch.show_sketch_folder = Προβολή φακέλου του Σχεδίου
+ menu.sketch.add_file = Προσθήκη αρχείου...
+
+ # | File | Edit | Sketch | Library | Tools | Help |
+ #                       | Library |
+ menu.library = Εισαγωγή Βιβλιοθήκης...
+ menu.library.add_library = Προσθήκη Βιβλιοθήκης...
+ menu.library.contributed = Συνεισφερόμενα
+ menu.library.no_core_libraries = αυτή η λειτουργία δεν έχει βασικές Βιβλιοθήκες
+
+ # | File | Edit | Sketch | Library | Tools | Help |
+ #                                 | Tools |
+ menu.tools = Εργαλεία
+ menu.tools.color_selector = Επιλογέας χρωμάτων...
+ menu.tools.create_font = Δημιουργία γραμματοσειρά...
+ menu.tools.archive_sketch = Αρχειοθέτηση Σχεδίου
+ menu.tools.fix_the_serial_library = Διόρθωση Σειριακής Βιβλιοθήκης
```






**Evolutionary System to Design Type**
Jéssica Parente, University of Coimbra, CISUC, DEI



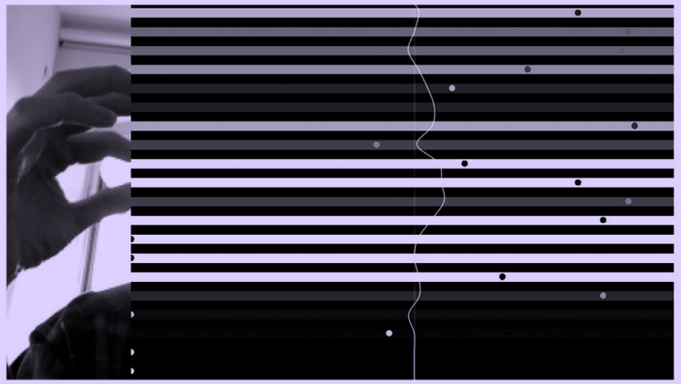
Happy 20th Birthday, Processing! Thank you for everything you do.

Devi Parikh
@deviparikh
#genartclub



# Scan Sequencer Javascript

JeongHo Park
https://jeonghopark.de
twitter@jeonghopark



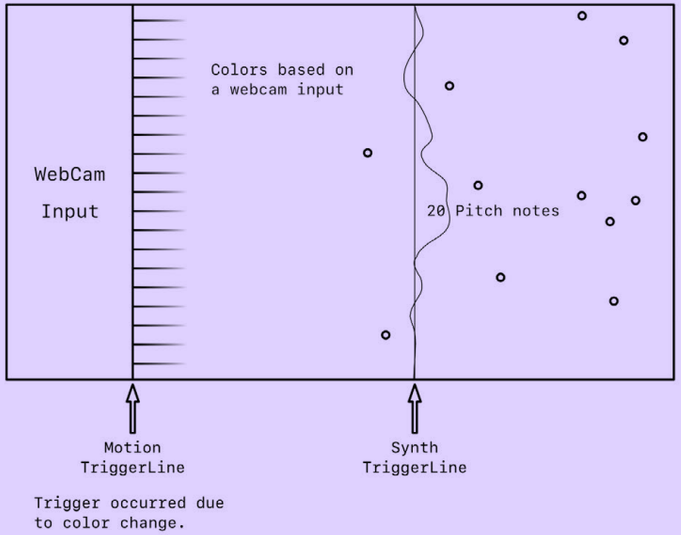
Scan Sequencer Javascript takes a realtime web cam image and in turn makes real-time music. It works like a virtual keyboard, the vertical keyboard is recognizing a change of the color of pixels and make a trigger signal. The circles made by trigger signals move to from left to right and when they cross the middle threshold, they play a note.

It is written based on the p5js and tonejs libraries.

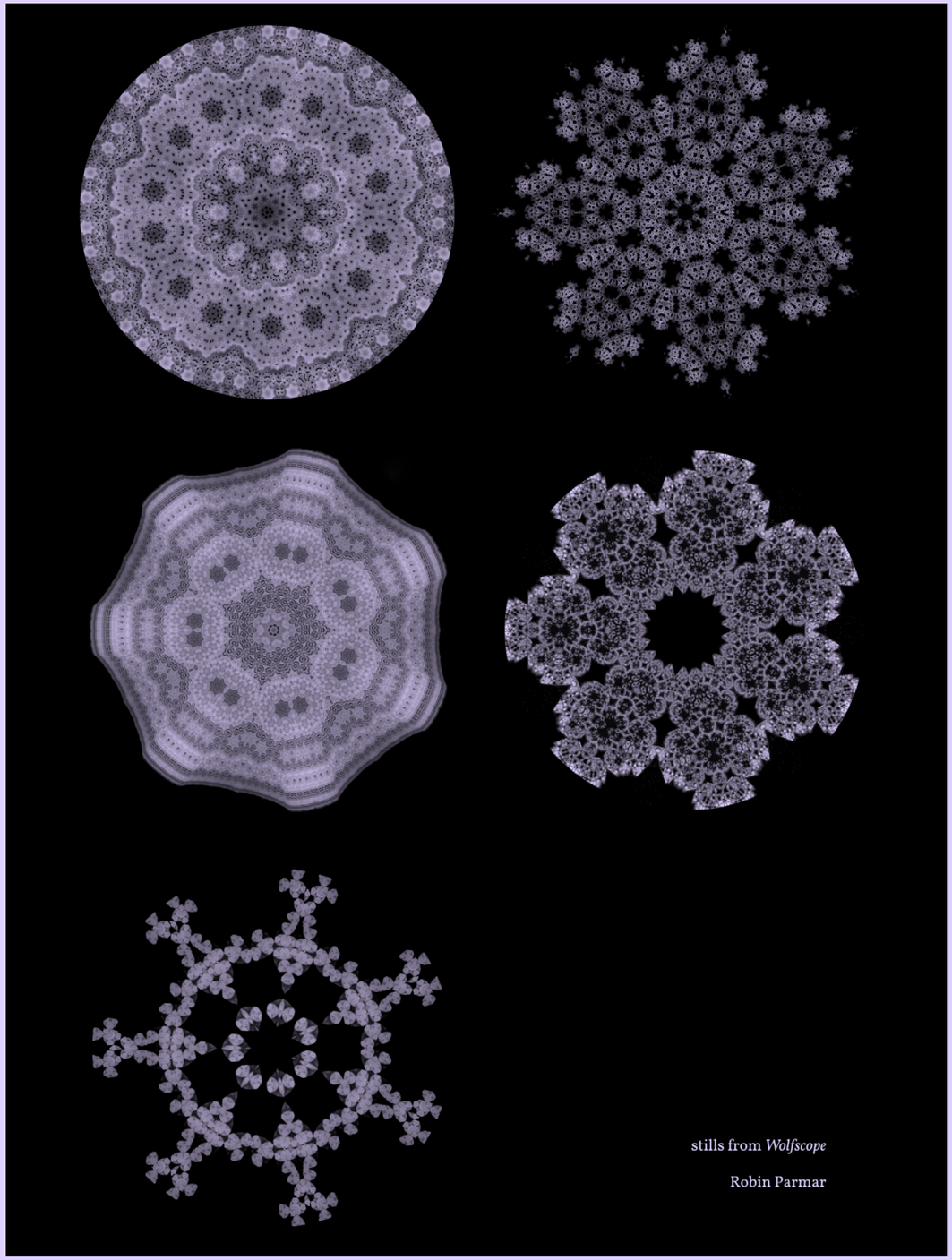
https://jeonghopark.github.io/scanseqjs/

https://experiments.withgoogle.com/scan-sequencer

https://jeonghopark.github.io/scanseqjs/




Photos - 10_PNG 95%
mindful_code | Processing 3.5.4
File Edit Sketch Debug Tools Help
Java
mindful_code
String mind = "present"; //initial state of mind
final static int NUMBREATHS = 10;
void setup() {
}
void draw() {
if (mind == "distracted") {
mindfulness("bodyscan");
} else if (mind == "monkeymind") {
mindfulness("count breaths");
} else if (mind == "anxious") {
mindfulness("deep breaths");
}
}
void mindfulness(String mind) {
if (mind == "bodyscan") { //passing different states of mind
//mentally scan body from top down to realign focus
Photos - 2.PNG 44%
안녕하세요
vintage_compArt_tut01
Photos - 10_PNG 116
metro 80
counter 48 162
+12 (next octave)
counter 60
mtof
phasor~ 220.
rect~ 440.
phasor~ 660
2.PNG


stills from *Wolfscope*
Robin Parmar

# P5 Turtle Microworld

**********

# Logo/Turtle Art Simulator in P5 by Cyberparra

```
function start() {

  var k=0;
  for(n=0;n<50;n++){
    t.forward(20+k);
    t.right(90);
      k=k+5;
      t.left(0.5);
  }
```





***After Fischli and Weiss,*** *2018.*
Video projection, discarded cemetery flowers, endoscopes, and Processing.

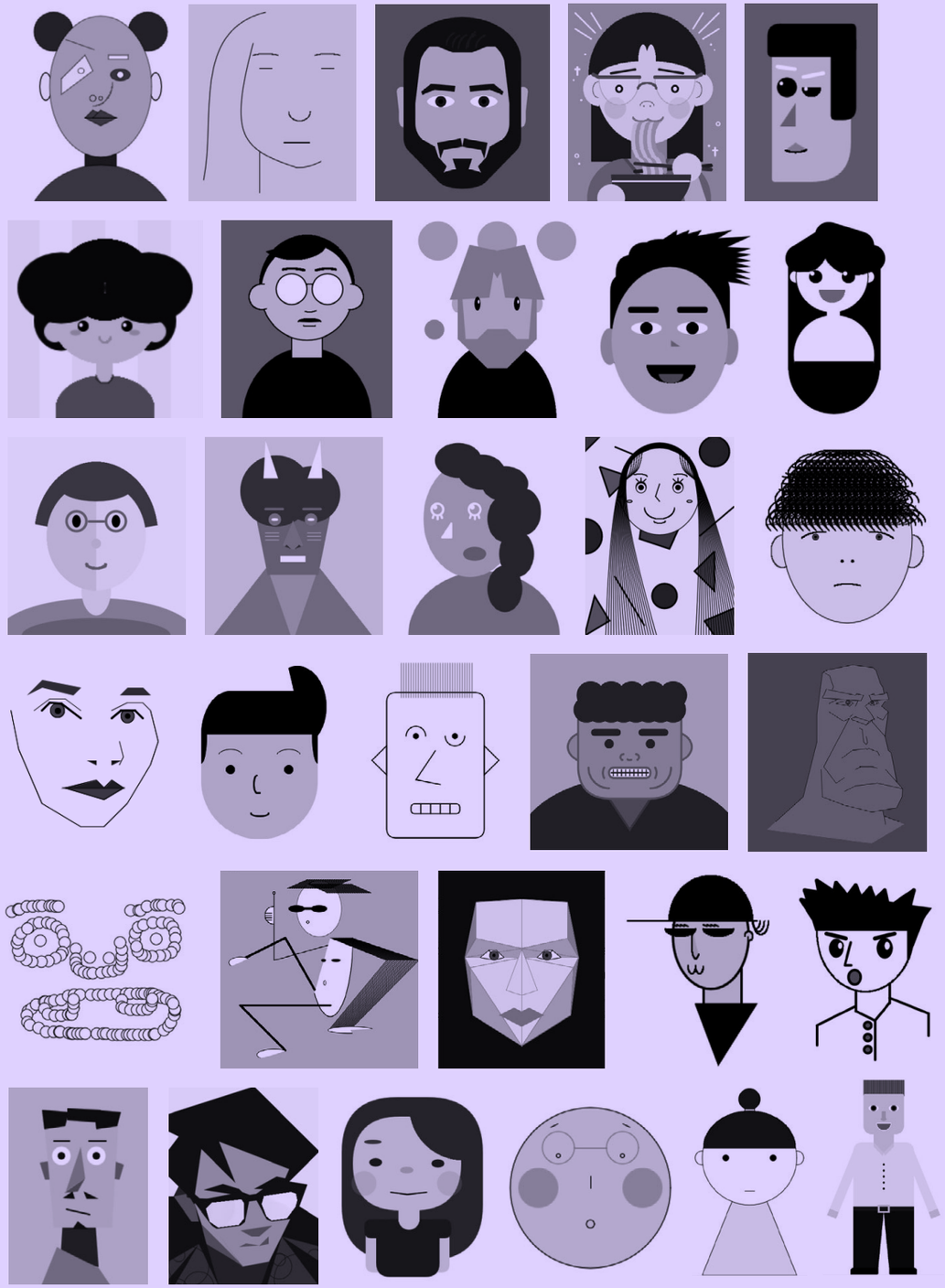
***3.20.18, 9:15 AM PST,*** *2018.*
Collaborative Video Project* and Processing.

***Dark Spring VR,*** *2018.*
Virtual Reality experience, ft. "Wake" by RYAT.

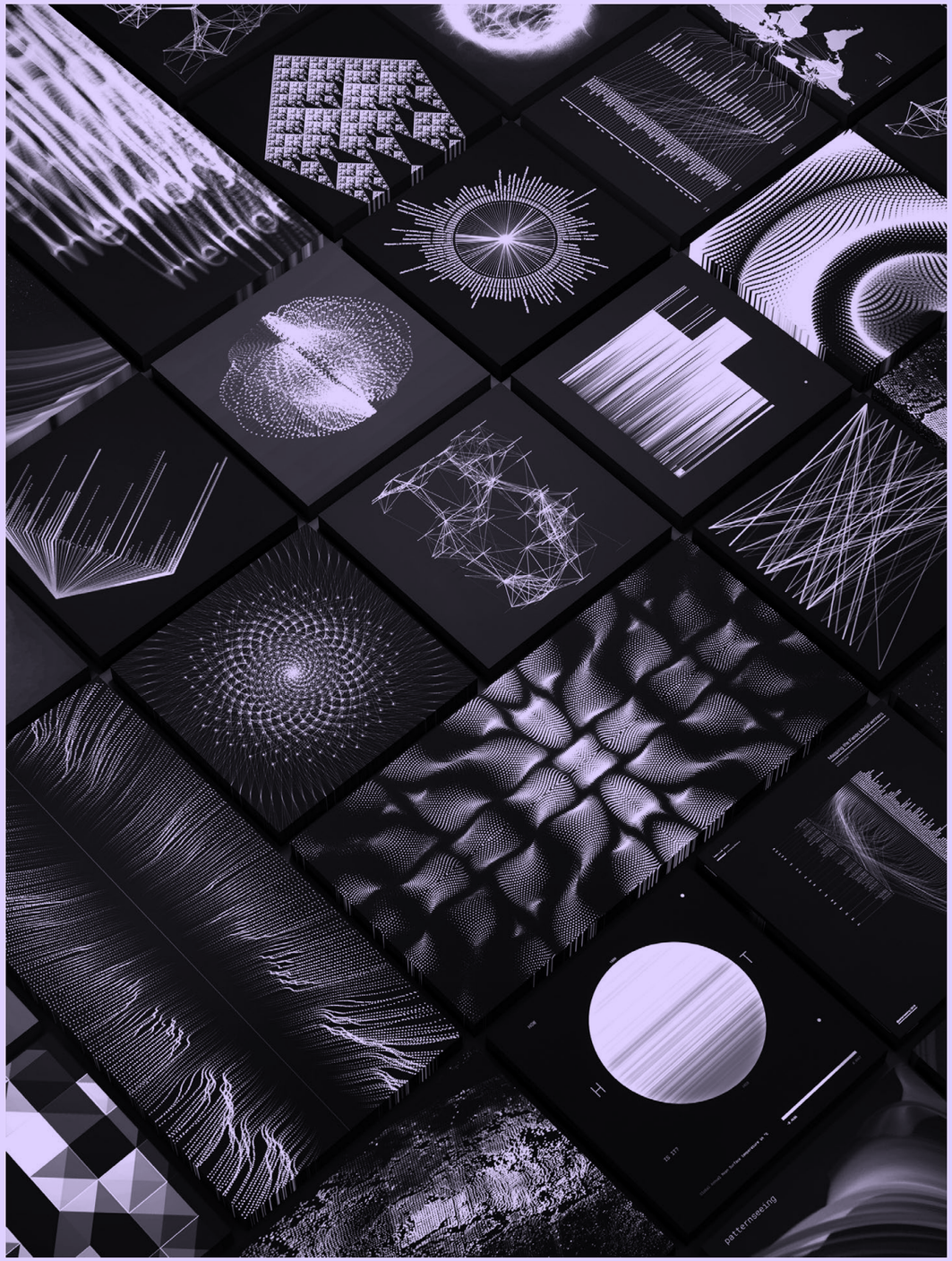
**3.20.18, 9:15 AM PST*
is a collaboration by Parsons and over 40 media artists from around the Los Angeles area, including Anna Luisa Petrisko, Candace Reckinger, Chandler McWilliams, Eileen Cowin, Ethan Gold, Eva Aguila, Gavin Gamboa, Jonathan Snipes, Laura Darlington, Lee Tusman, Michael John Kelly, Natalie Sun, Nathaniel Eras, Nina Sarnelle, Peter Flaherty, Rachel Binx, Samantha Culp, Sarah Zucker, Sharsten Plenge, Ty Pownall, Victoria Vesna. Video taken at the exact moment of the vernal equinox was remixed into a collage using Processing.

As the culmination of a media arts residency at Coaxial Arts, media artist Kate Parsons unveiled new video work, a virtual reality prototype, and floral installations comprised of fresh florals and discarded cemetery flowers.

This residency was made possible by a grant from the Pasadena Arts Alliance, Coaxial Arts Foundation and technical support from FLOAT.






patternseeing

Fukushima Offset

2021, Mou Peijing Melody

Processing / Python / Physical computing

Details can be found here: https://www.melodymou.com/fukuoffset

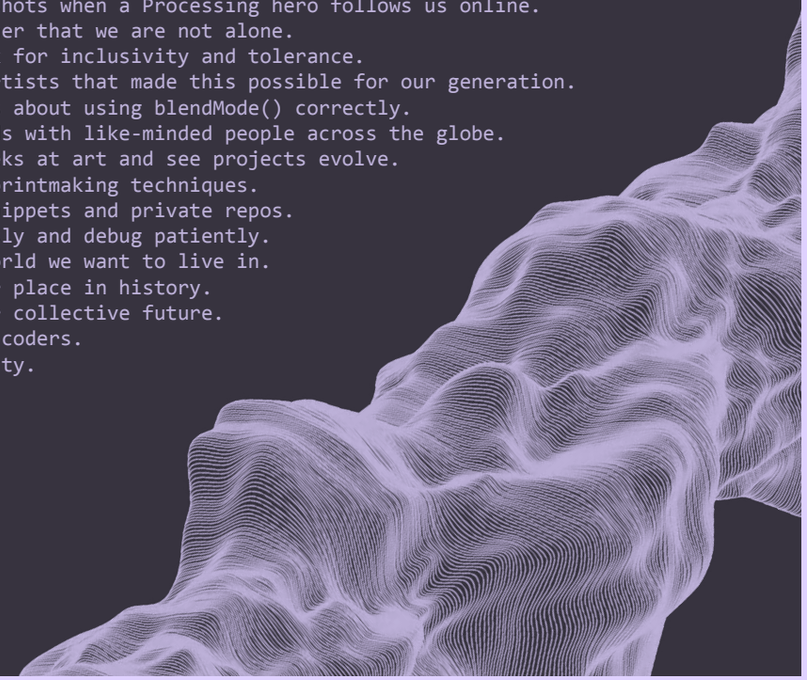


Creative coding is all about the community.

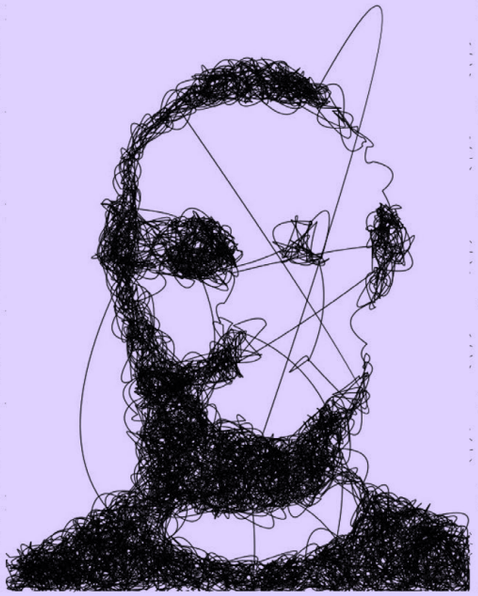
So much artistic beauty comes from people who express themselves with code, but by connecting, by being open with each other, we create something even more meaningful than the artwork.

We push our craft forward.
We inspire each other and share research.
We encourage and lift each other up.
We critique colors.
We send pictures of our dogs.
We share mental health struggles.
We brainstorm new ideas.
We repair each other’s quadtrees.
We deepen our understanding of diversity.
We share pricing and contract information.
We learn from each other’s successes and failures.
We get jealous.
We celebrate.
We create beauty.
We organize meetups.
We plan tech/art startups.
We teach new artists old tricks.
We invite people to explore.
We debate ways to slow climate change or improve our homes.
We share screenshots when a Processing hero follows us online.
We show each other that we are not alone.
We actively work for inclusivity and tolerance.
We revere the artists that made this possible for our generation.
We ask questions about using blendMode() correctly.
We become friends with like-minded people across the globe.
We get sneak peeks at art and see projects evolve.
We explore new printmaking techniques.
We share code snippets and private repos.
We give selflessly and debug patiently.
We create the world we want to live in.
We establish our place in history.
We determine our collective future.
We are creative coders.
We are a community.

Aaron Penne
@aaronpenne
#genartclub



# Thank you Processing



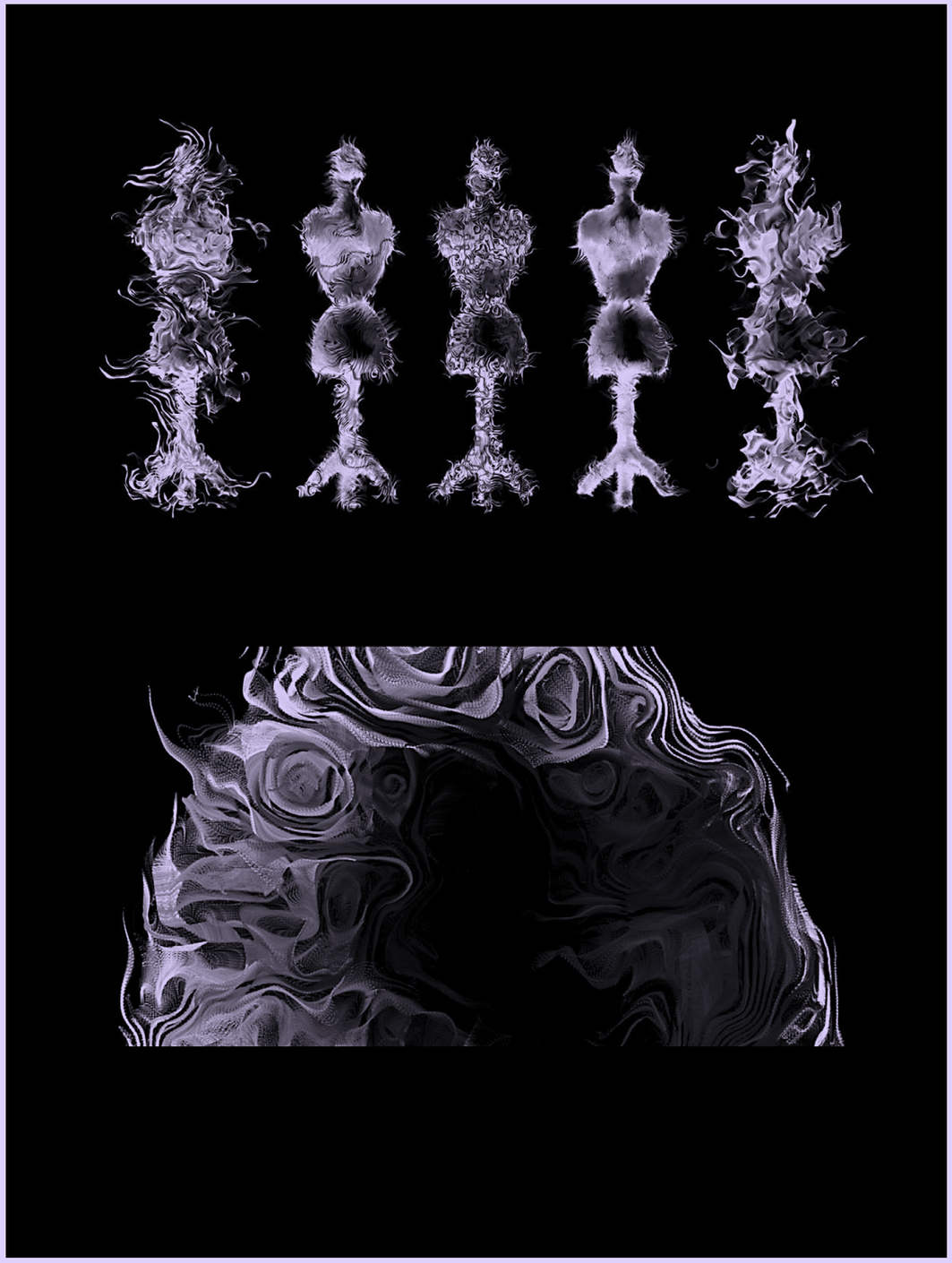
Thank you *Daniel* for being the teacher I never had and the inspiration for who I wanted to be.

Thank you *Casey* and *Ben* for creating the tool that allowed me to express my creativity.

Thank you *Lauren* for always being an inspiration with all your projects.

Thank you to the *entire community*. I hope I have given back at least some of what you have given me all these years.





## Generous Art

Phoenix Perry, 10/2021

Some journeys start with a hello. I meet Casey at his Bitforms show in New York in February of 2002. His generative art covers the walls. It is created in using an open source tool he made called Processing. Organic flowing lines spin paths of colour that fill the gallery walls. Next to the pieces hangs their code - naked, exposed, and made vulnerable. His process is laid out for anyone to use, improve, or understand. The work felt transgressive in its generosity. The altruism was contagious.

This kindness drew me to working with Processing. As soon as I have some free time, I download it and start coding. Mailing lists quickly pop up. I make code experiments and indie games. Whenever I start something new, Processing constantly forms the initial investigation.

To support the community, I open Devotion gallery with computational biologist Marie Evelyn, sound artist Margaret Schedel, and musicians Brian Jackson and David Last. Between 2009 and 2014, we host close to 300 events and exhibitions. Many of the artists we show experiment with Processing. It is obvious this is tool shaping a generation of creators. As the years move on, I begin to notice a severe lack of diversity in the scene.

At a conference that I present at in 2011, I am shocked by the state of affairs in the field. I show my interactive machine learning work done in collaboration with Dr Rebecca Fiebrink and Dr Margaret Schedel. It was the very first work of its kind. Many men around me are so wholly unable to see me as a peer that they mostly ignore me. To make matters worse, I am sexually harassed by prominent community members. The conference intentionally welcomes marginalized creators, but the discipline required more than our presence to make us safe. It needs our leadership.

It would be remiss of me not to point out many of the key players in this history of Processing, and the creative coding community, are in the Global North. Also, it is important I acknowledge my own whiteness. While I do live with a disability, I'm privileged. Would these spaces have been as accessible to me otherwise?

In 2012, I quit my day job in design and move to teach at NYU. I am asked to join by two artists whom I know from the Processing Community. On my first day at ITP, the NYU receptionist leads me to the sewing machines when I say I am teaching and need help finding my classroom.

In March of 2013, I take action to bring change. I reach out to three women and they are Catt Small, Jane Friedhoff, and Nina Freeman. We bring Code Liberation to life. We run free queer-friendly coding classes only for women and use them to build community and solidarity. Around 90 women show up to learn to code our first class. How could there be so much demand and so little support? Over the coming years, we teach several thousand women and non-binary people to code for free. During this period, Lauren McCarthy is building p5. Evident to us both, there exists an affinity between projects, and Code Liberation begins teaching p5.

By 2015, I leave NYU and am at Goldsmiths in London. Code Liberation Foundation collaborates with more women, including Saskia Freeke, Adelle Lin, Alice Casey, Charlie Ann Page, Kiona Niehaus, and Caroline Sinders. Saskia delivers a series of p5 classes during 2017 culminating in an exhibition of 13 games at the V&A. This project also generates a set of open source teaching materials for p5.

Starting in late 2018, I transition to UAL. Where I work to develop the curriculum for the Creative Computing Institute. Interlaced in our work are principles of social justice. Given you can get a degree in it, is creative coding still radical? How do tools like Processing survive when Universities that teach with them largely don't contribute to their survival? Most of the key figures in this community now work within academia. Have we been co-opted? Is there another way? These are open questions.

What is Processing for me? It has been the yarn weaving my creative and professional lives together, and without it, I'm not sure this sweater would have turned out quite the same.



# Processing Foundation Fellowship

## Niklas Peters, 2017 Fellow

## Developing a creative coding curriculum for learners with low computer literacy

The Processing/p5.js community has generously created countless resources for people to teach themselves to code. However, many people face significant barriers in accessing these self-directed resources. Learners from marginalized groups can lack a baseline familiarity with computers, and even if they become more comfortable with computers, coding can seem cryptic and inaccessible.

My fellowship aimed to bridge that gap. I designed a curriculum for students with low computer literacy, with the end goal of having participants make basic programs and feel comfortable to continue their coding education by accessing free resources on their own.

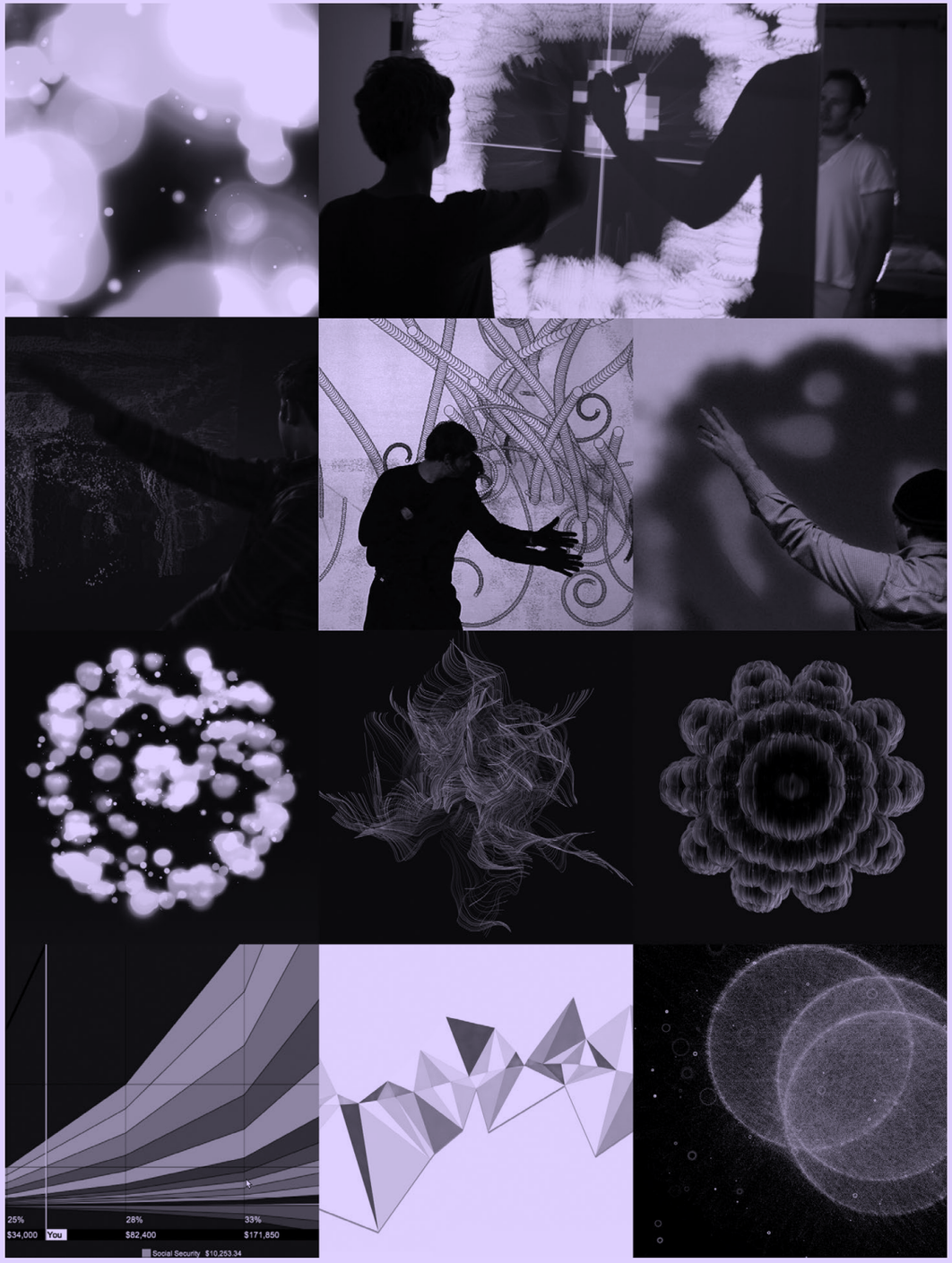
In July 2017, I ran a pilot of my curriculum with high school students in Johannesburg, South Africa. The first few modules covered the basics of using a computer, including drawing with different brushes and colors in MS Paint, exploring keyboard shortcuts and special characters, and navigating files and folders in Windows.

The most exciting part of the pilot was seeing the students create their first sketches in the p5.js web editor. There were murmurs of excitement throughout the computer lab as they saw their first circles appear on the screen, and found they could control a circle's size and position with tiny modifications to the code. As the students learned more about colors, variables, and how to make their sketches interactive, their excitement about the possibilities of creative coding grew.

I continued to run the program throughout 2017 with other after-school groups in Soweto, making adjustments to the curriculum as I learned from the students which exercises were most useful in mastering the basics.

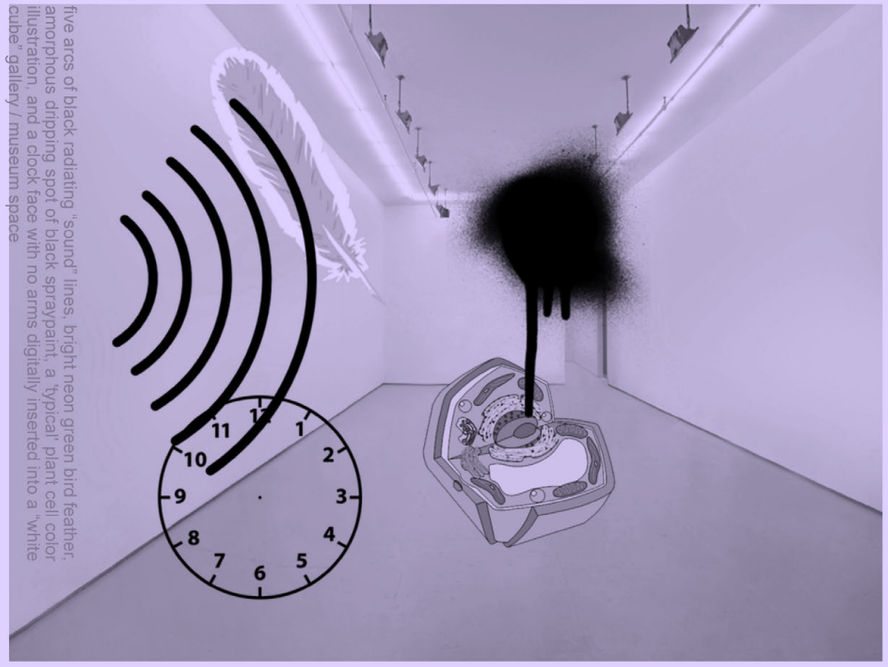
The 10-module curriculum is available for use & modification:
https://github.com/nikfm/p5jsCurriculum

Niklas Peters is an artist, musician, and organizer
More info at http://niklaspeters.com






stone/boulder (reddish-brown, realistic) with roots (black & white illustration) growing from it, tan and brown animal fur, slanted-style bonsai tree, and a light gray plastic bag digitally inserted into a "white cube" gallery / museum space



five arcs of black radiating "sound" lines, bright neon green bird feather, amorphous dripping spot of black spraypaint, a typical plant cell color illustration, and a clock face with no arms digitally inserted into a "white cube" gallery / museum space

Noah Travis Phillips / @ th3n04h
NoahTravisPhillips.com

*Phenomenology & New Materialism(s) exhibition rehearsals*

…is one of the first projects I ever made in Processing. I couldn't have done it without the community. My practice is collage/montage based, and because of the substantial and generous documentation as well as forums and friends, I could apply these same creative and conceptual approaches to coding.

I pieced the code together through multiple iterations, understanding the language and logic better with each version, and becoming more articulate with what and how I was able to say things with the program(ming), adapting it in dialogue with my aesthetic, ideas, and praxis.

# Hi there.

## I'm Nikko.

Processing is just amazing.
It's the way I first fully grasped programming.
It's just so fun to be able to make cool art with text.
Not to mention the fact that you could make the art so interactive
to the point it turns into a game. It's just so cool.

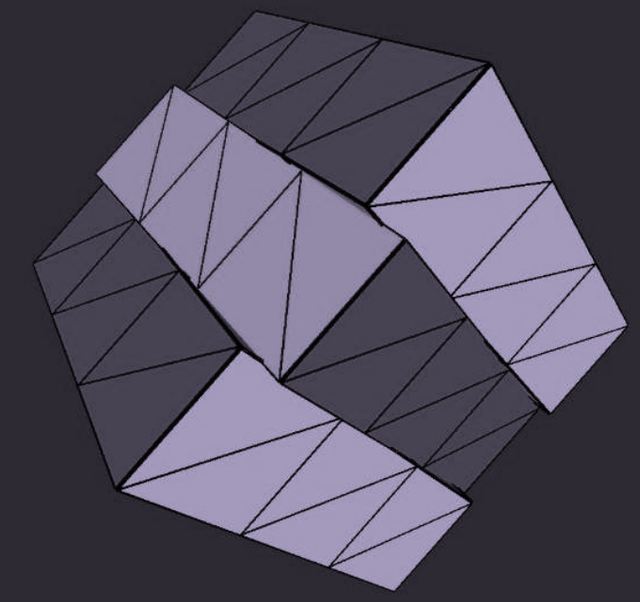
Knowing how to code opened up a lot of possibilities for me
and I had a lot of fun just messin' around and making weird shapes.
It's so awesome that anyone can use Processing
and have as much fun as I did.

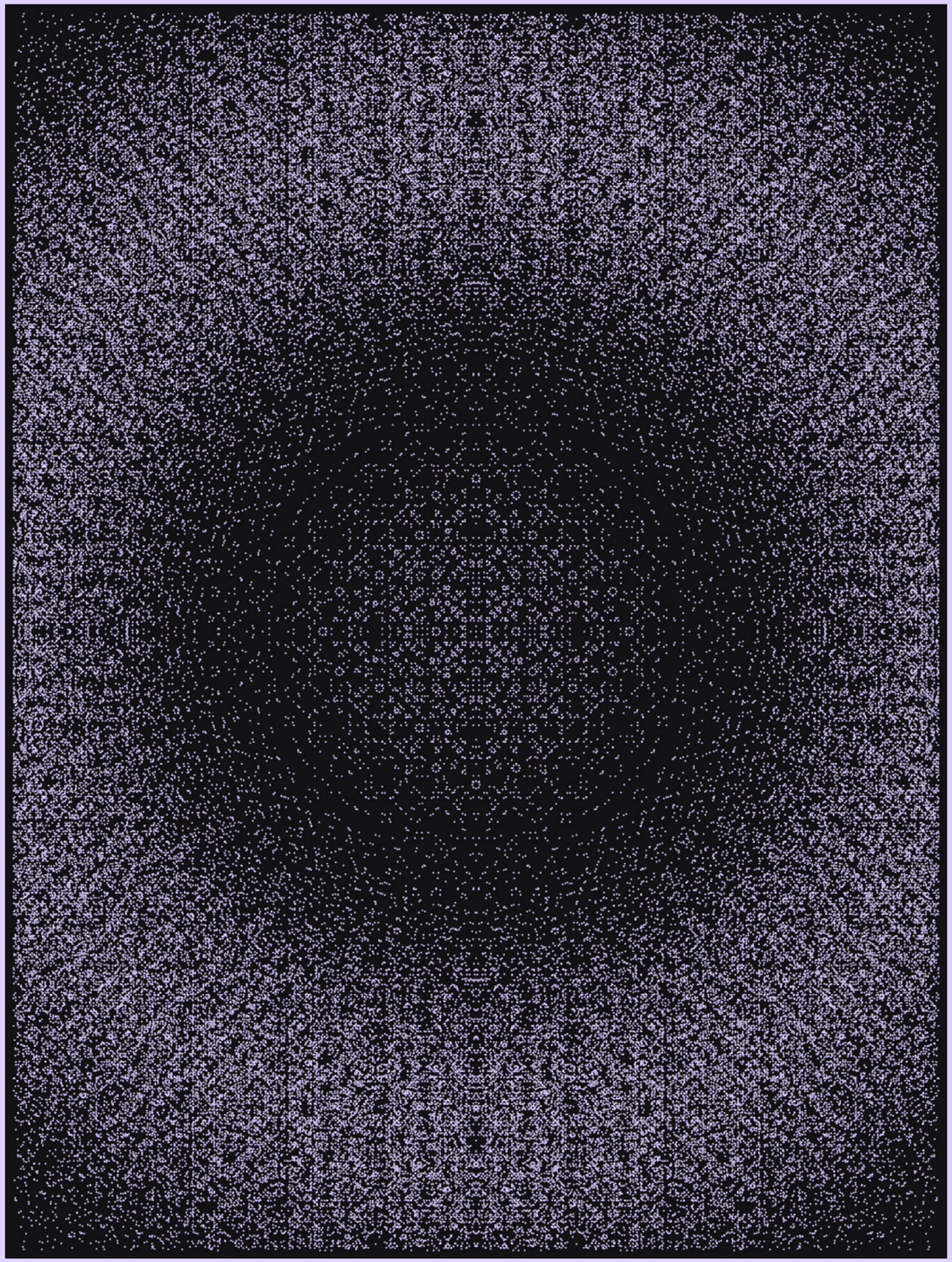
Thank you Processing :D


```
let boxSize = 80;
let boxR = [];

function setup() {
  createCanvas(500, 500, WEBGL);
    ambientLight(255);
    for(let i = -boxSize; i <= boxSize; i+=boxSize){
        let cosa = [];
        for(let j = -boxSize; j <= boxSize; j+=boxSize){
            let cosa2 = [];
            for(let k = -boxSize; k <= boxSize; k+=boxSize){
                cosa2.push(new Box(i, j, k, boxSize));
                //boxR.push(new Box(i, j, k, boxSize));
            }
            cosa.push(cosa2);
        }
        boxR.push(cosa);
    }
}

function draw() {
  background(51);
    rotateZ(frameCount*0.01);
    for(let i = 0; i < 3; i++){
```

Console

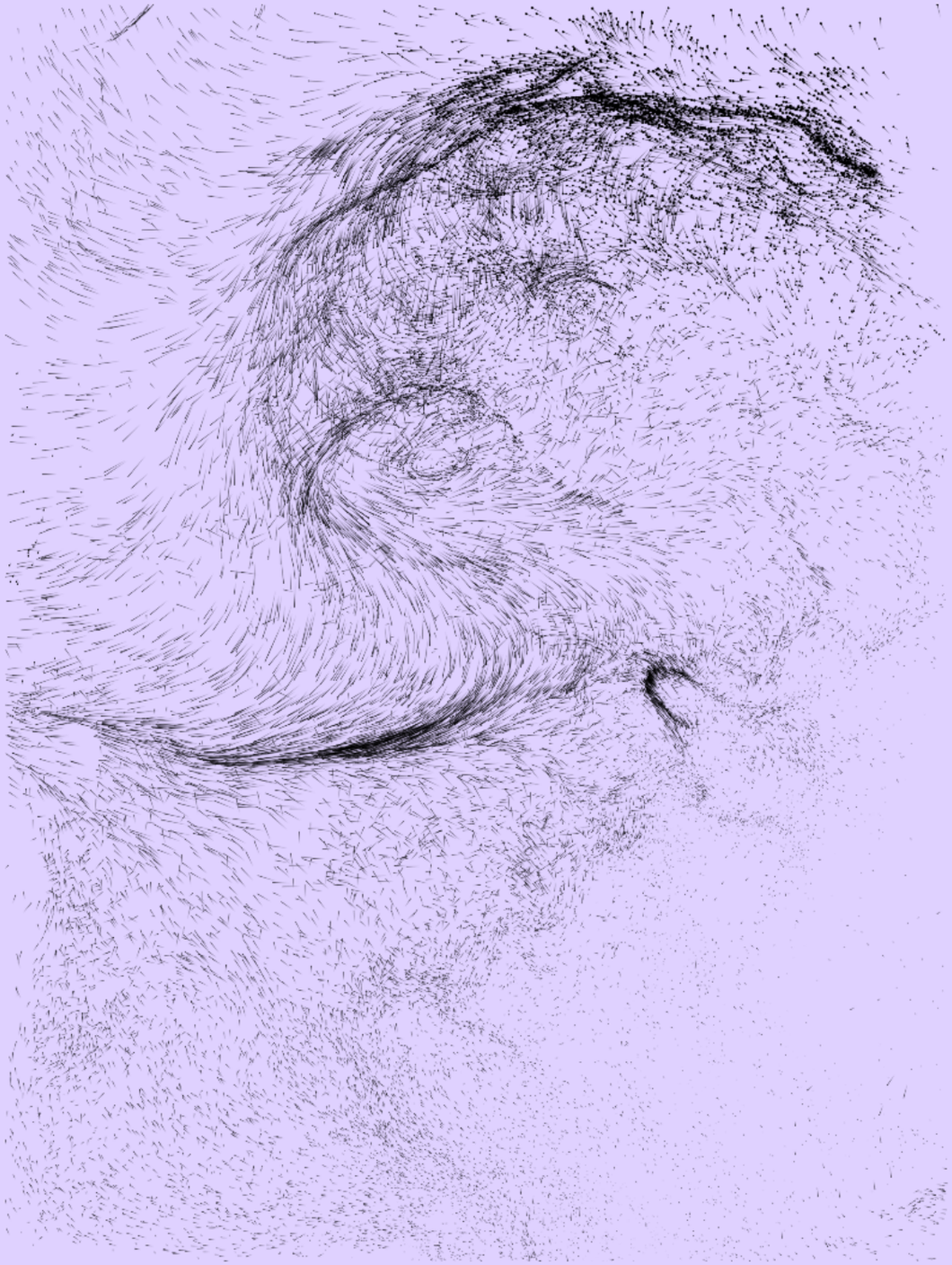
```
p5.RendererGL: enabled webgl context
```



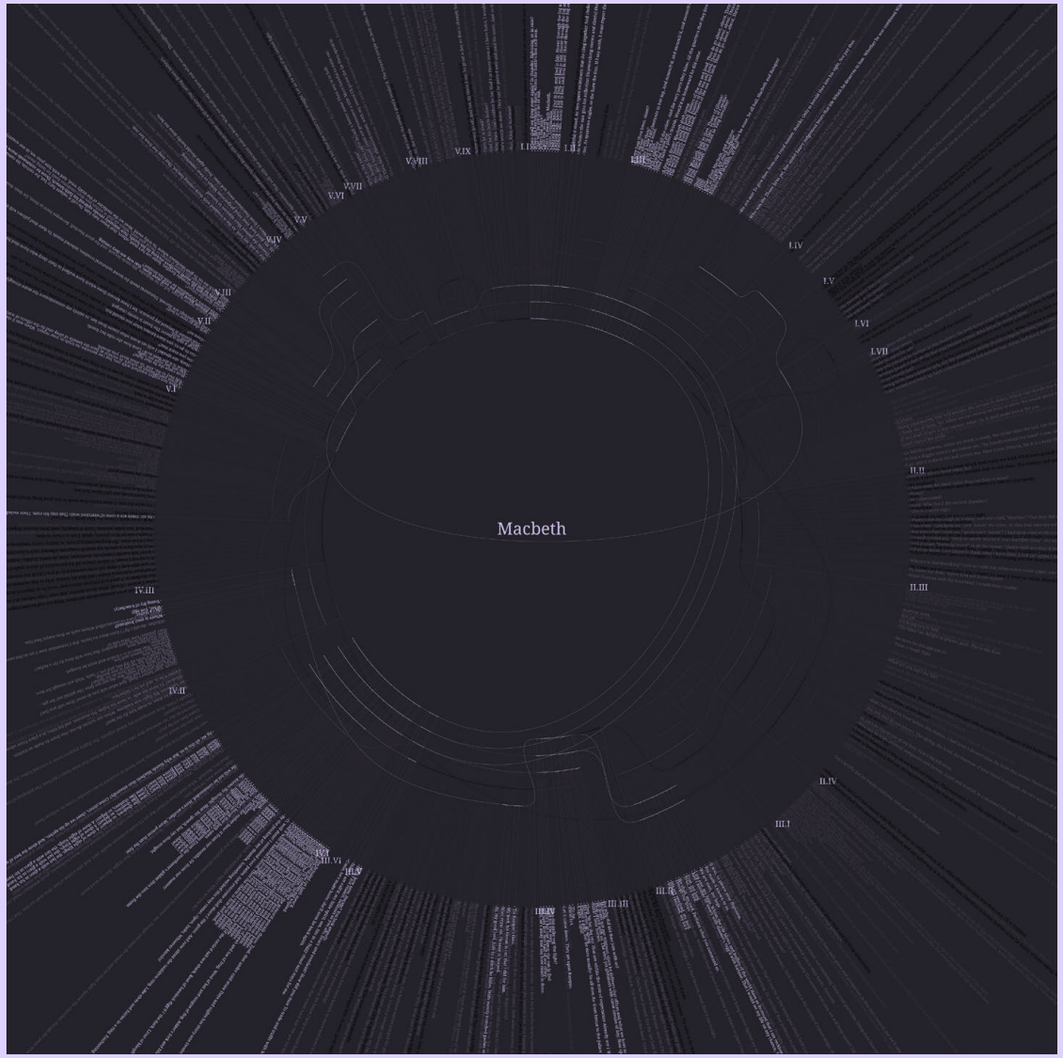


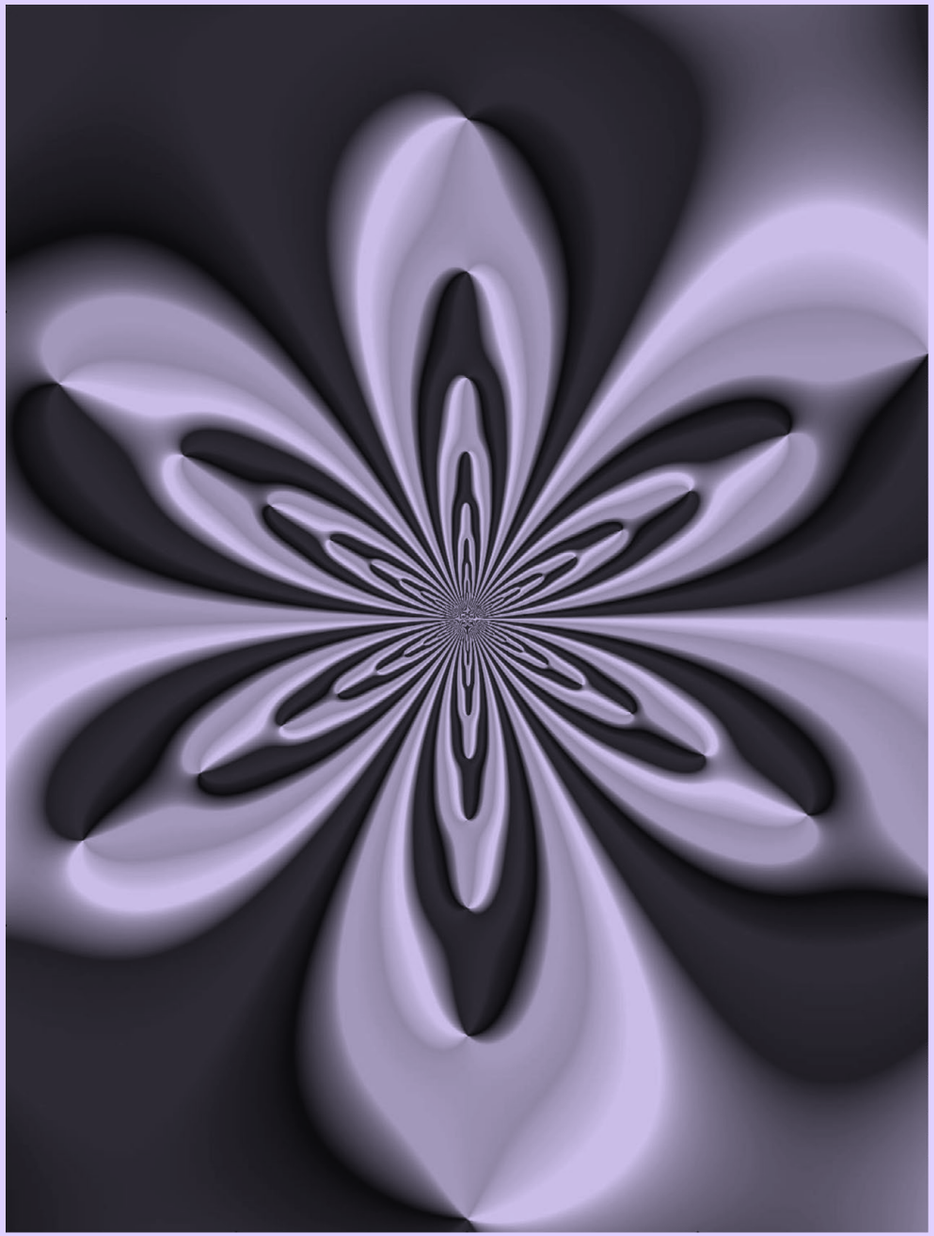



It all started with a sketch...






Macbeth

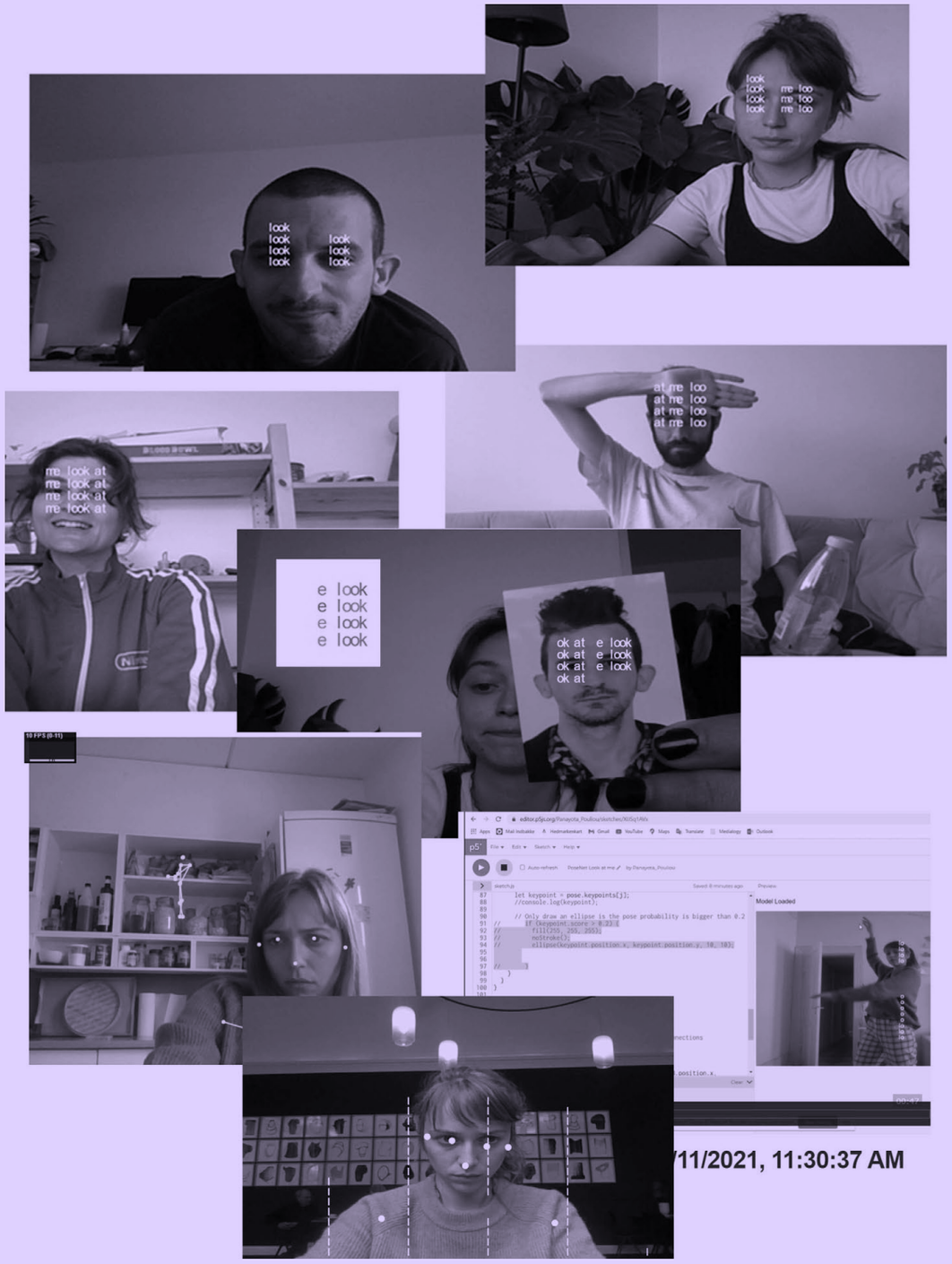


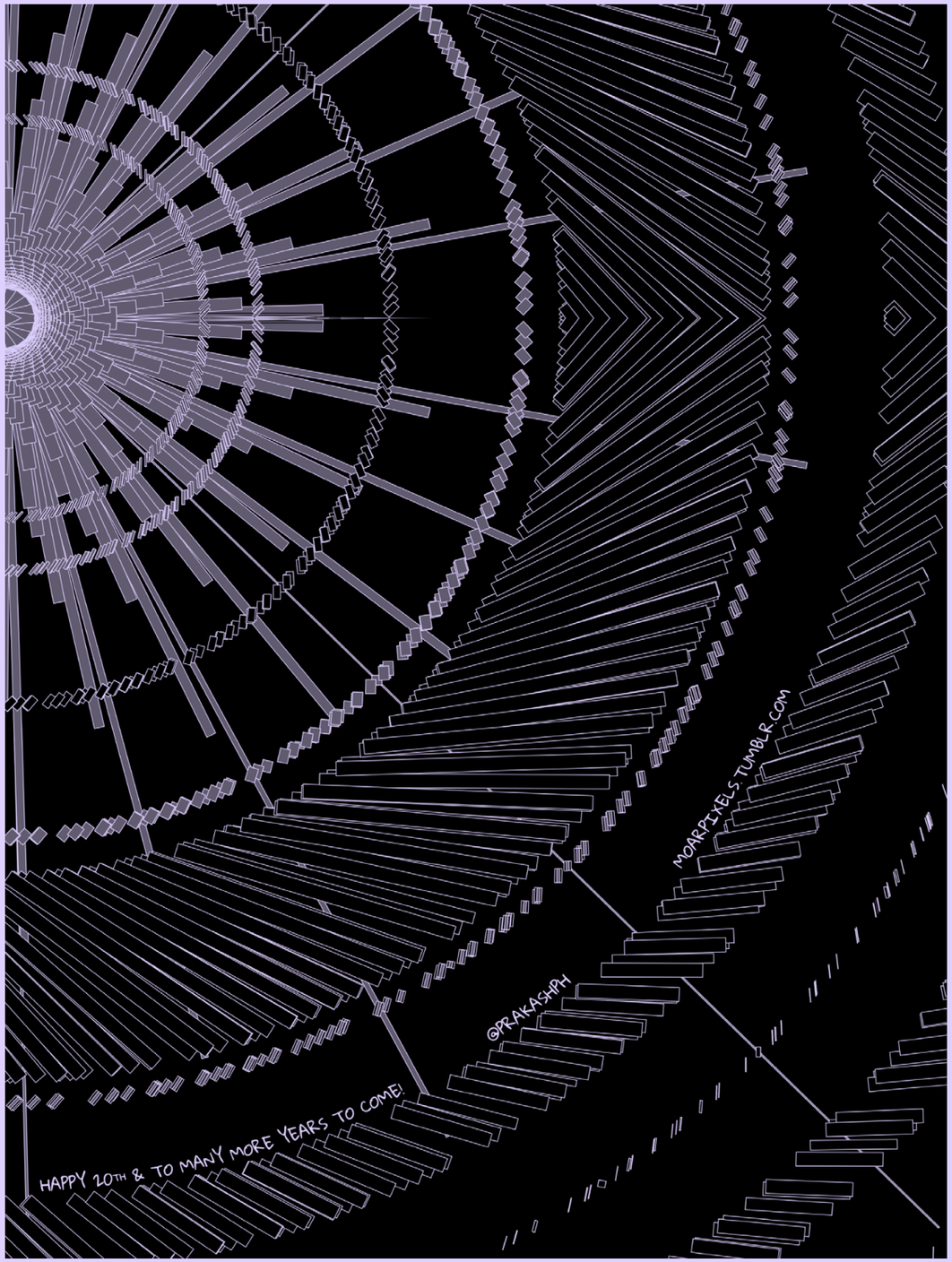
**The beauty of complex functions**

**@jcponcemath**

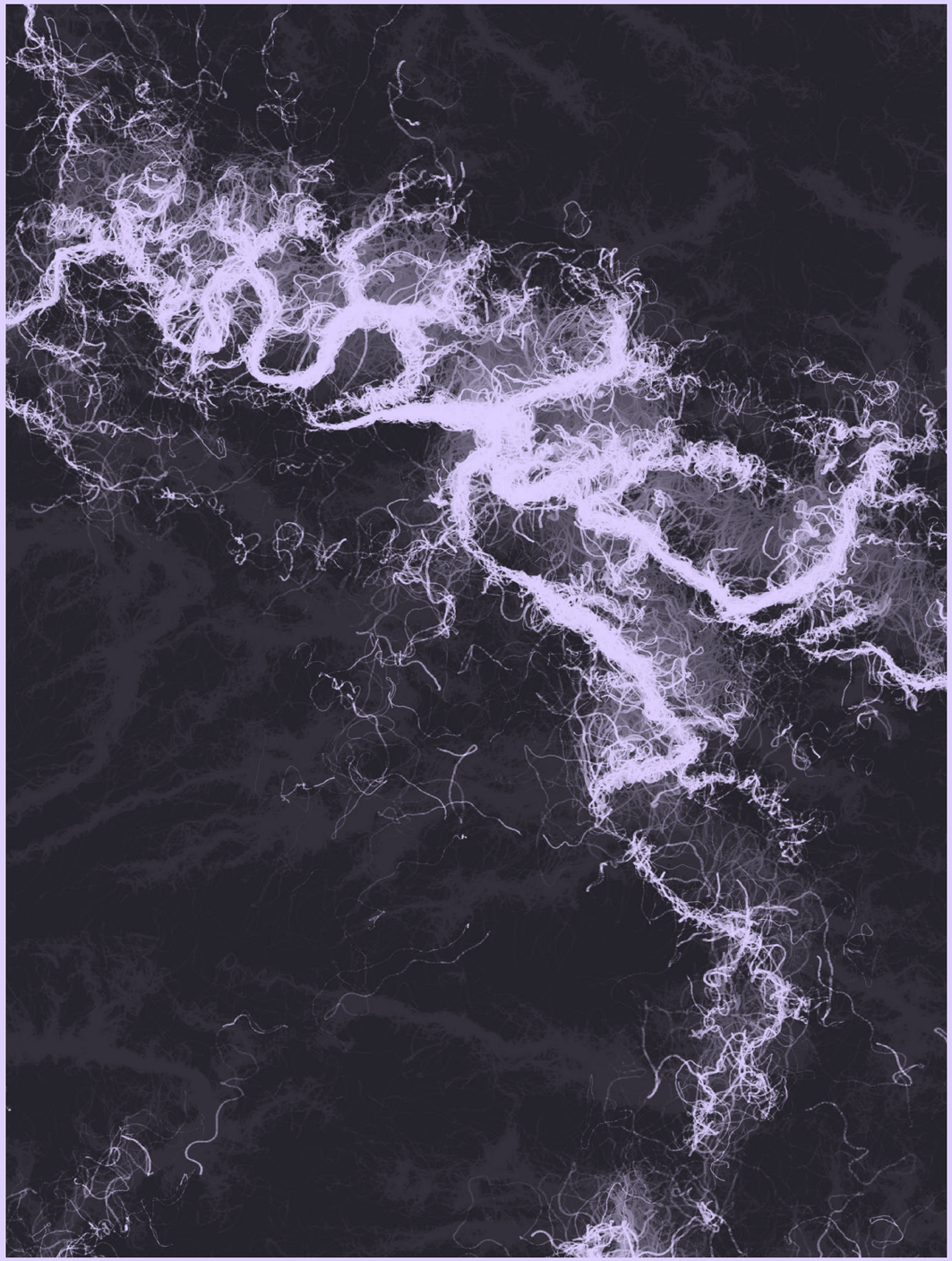
https://www.dynamicmath.xyz/domain-coloring/






HAPPY 20TH & TO MANY MORE YEARS TO COME!
@PRAKASHPH
MOARPIXELS.TUMBLR.COM





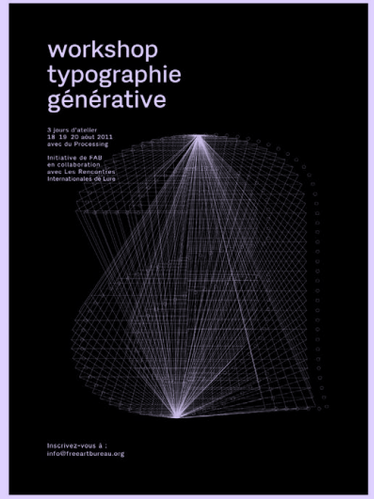




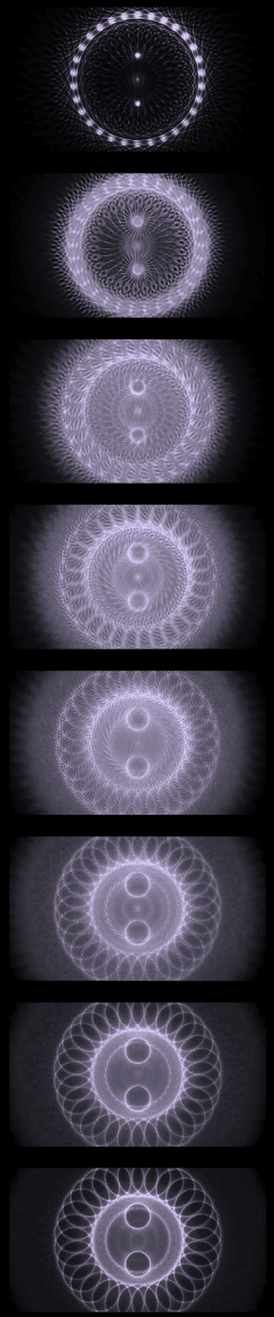

## PROCESSING PARIS 2010 - 2015 VARIOUS POSTERS FOR WORKSHOPS

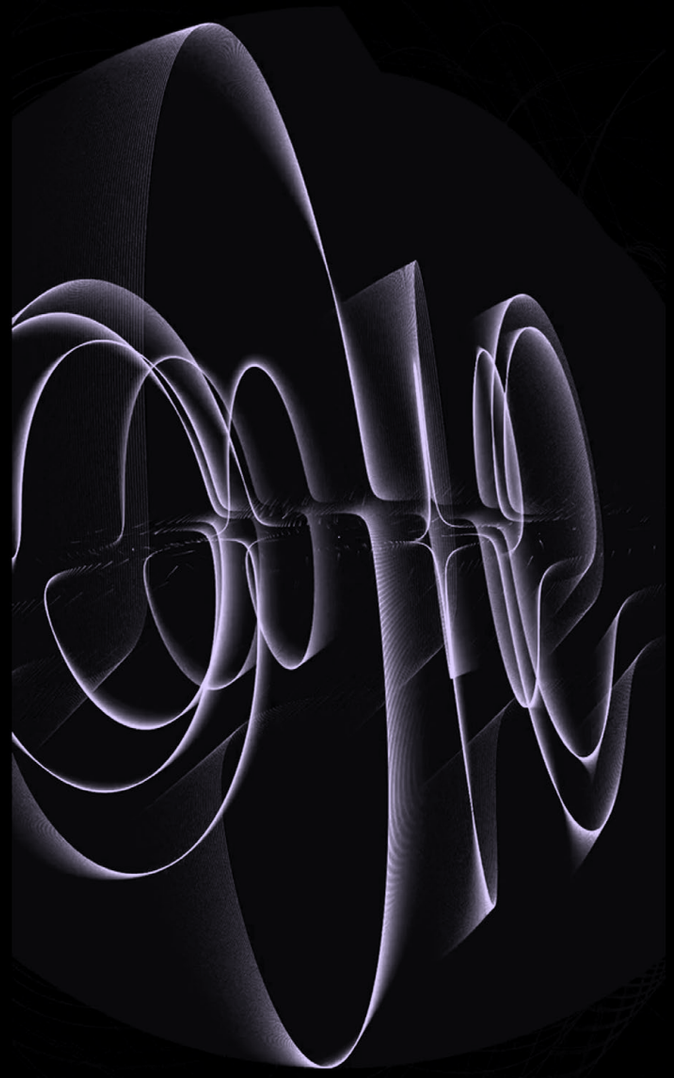






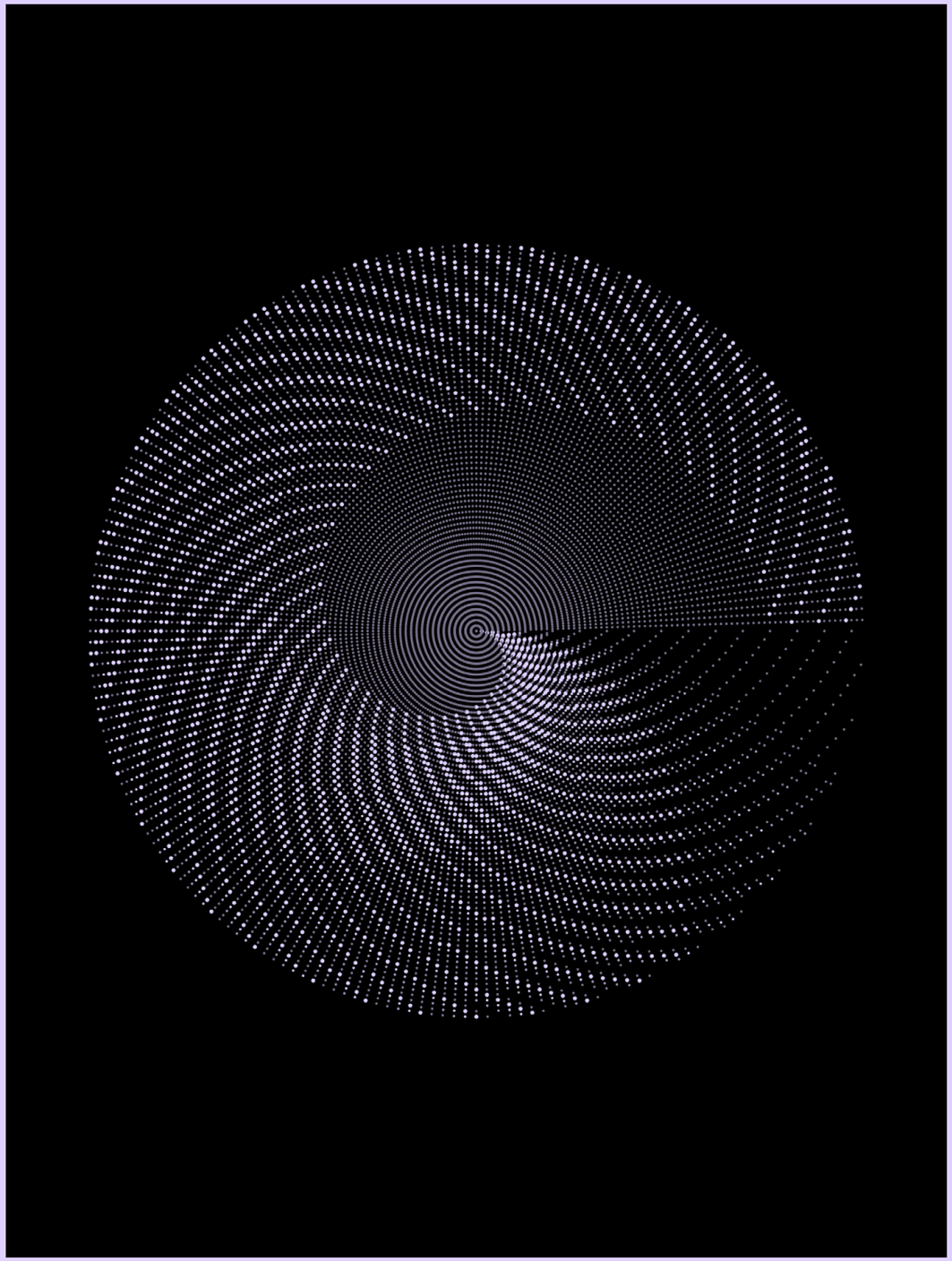

artwork & design by mark webster



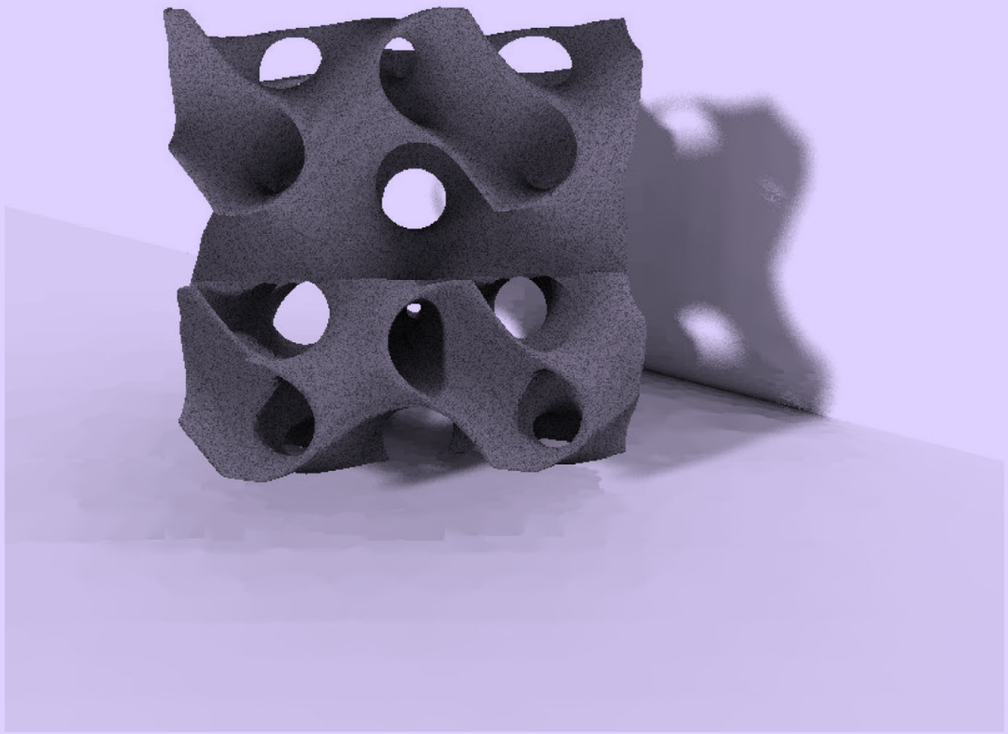


For me Processing is about the feedback loop. The engrossing quality of making changes and seeing their effects. It's how I learned to code, it shaped my practice, and the community has been there at every step. I started out by copying directly from examples created by the community. Then I tried to imitate others by trial and error. Finally I shared my own art and code. Then we went around again.



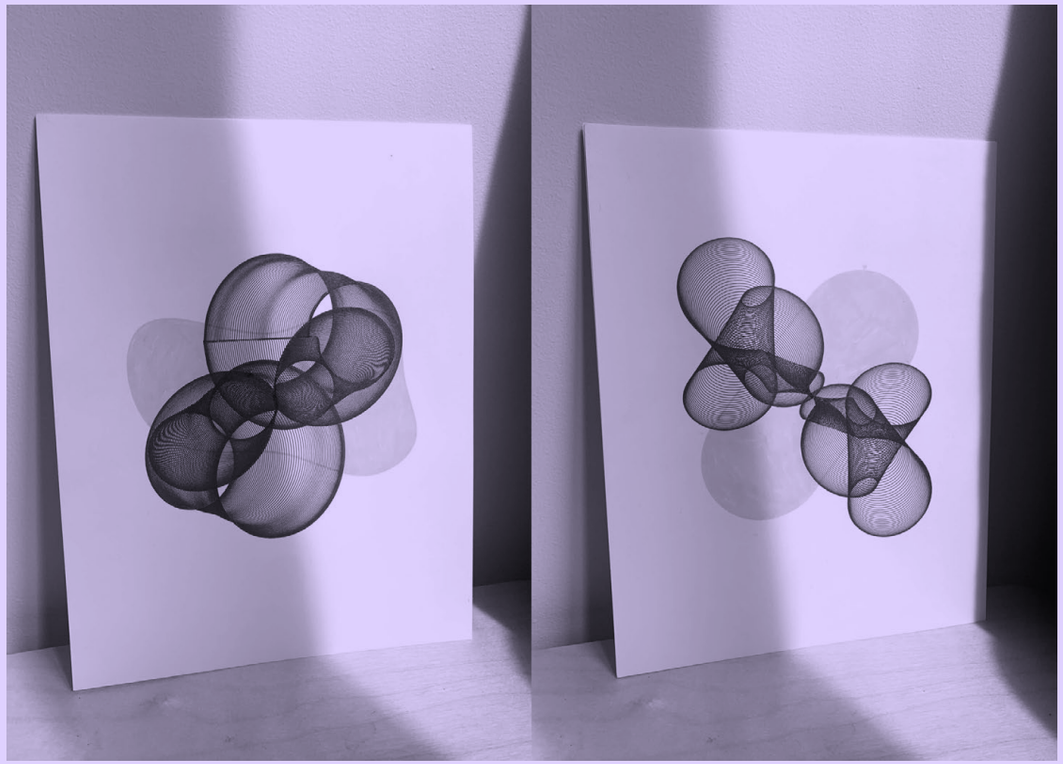


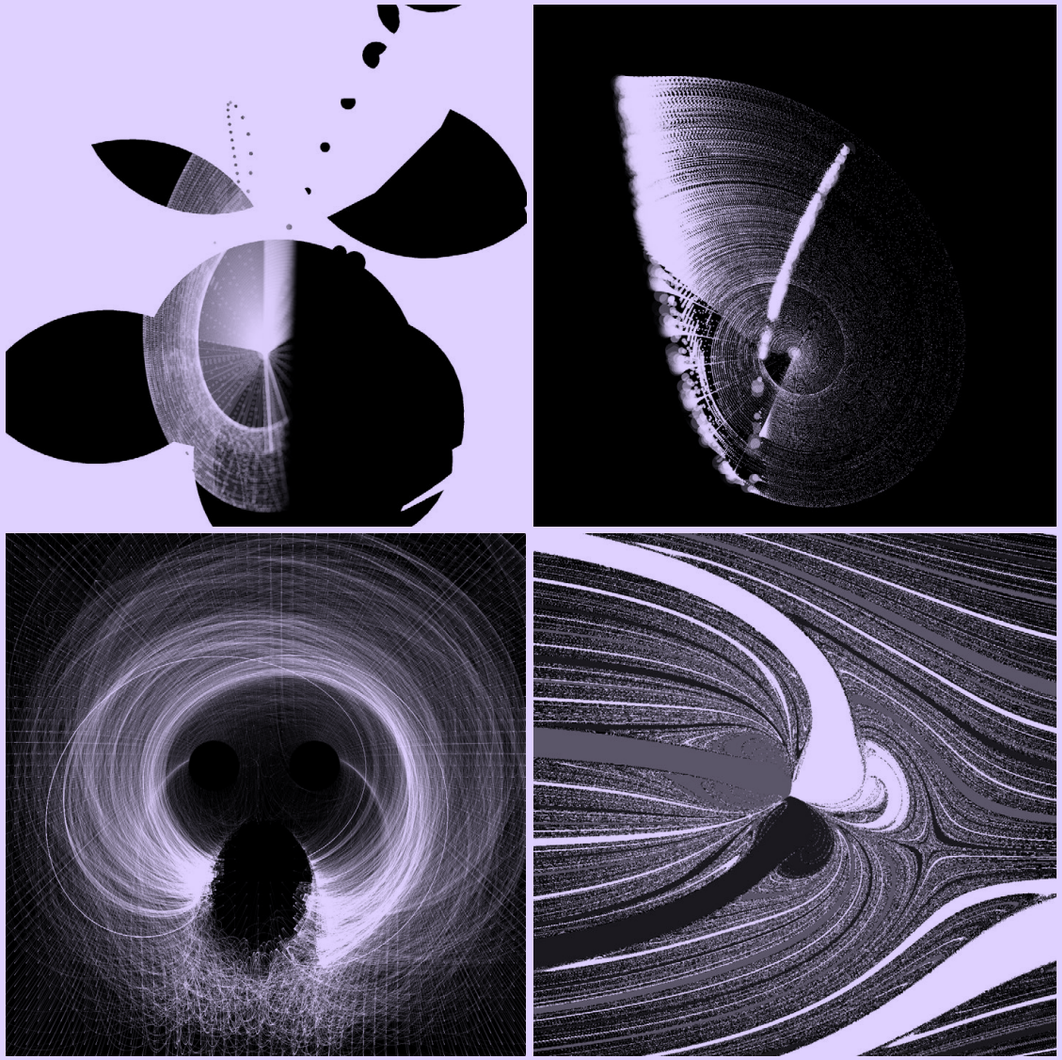
My name is Martin Prout aka monkstone (monkstoneT on twitter).
In 2008, when Darrel Ince was professor of computing at the Open University (Milton Keynes), Darrel created a Mass Writing Project for students and former students, and I was one of them. As an enticement and inspiration we all received a copy of Ira Greenbergs Processing book, and were encouraged to start a blog (which I have since deleted). By 2009 I was teaching myself ruby, so that I could explore ruby-processing. Sometime later Jeremy Ashkenas (whilst on one of his extended bike tours) granted me write access to his ruby-processing github project, which I maintained and updated for processing-2.0. For processing-3.0 I created JRubyArt, which uses some vanilla processing code, but is no longer dependant on a vanilla processing installation, and includes some novel extras. I have also made some processing libraries available as rubygems, notably toxiclibs, geomerative, wordcram and pbox2d. I have had spells of exploring LSystems in java, ruby and python. I have also explored ray-tracing using both sunflower (java) and povray (C++) in combination with processing. Here is a povray example using a mesh exported from toxiclibs (run from JRubyArt):-



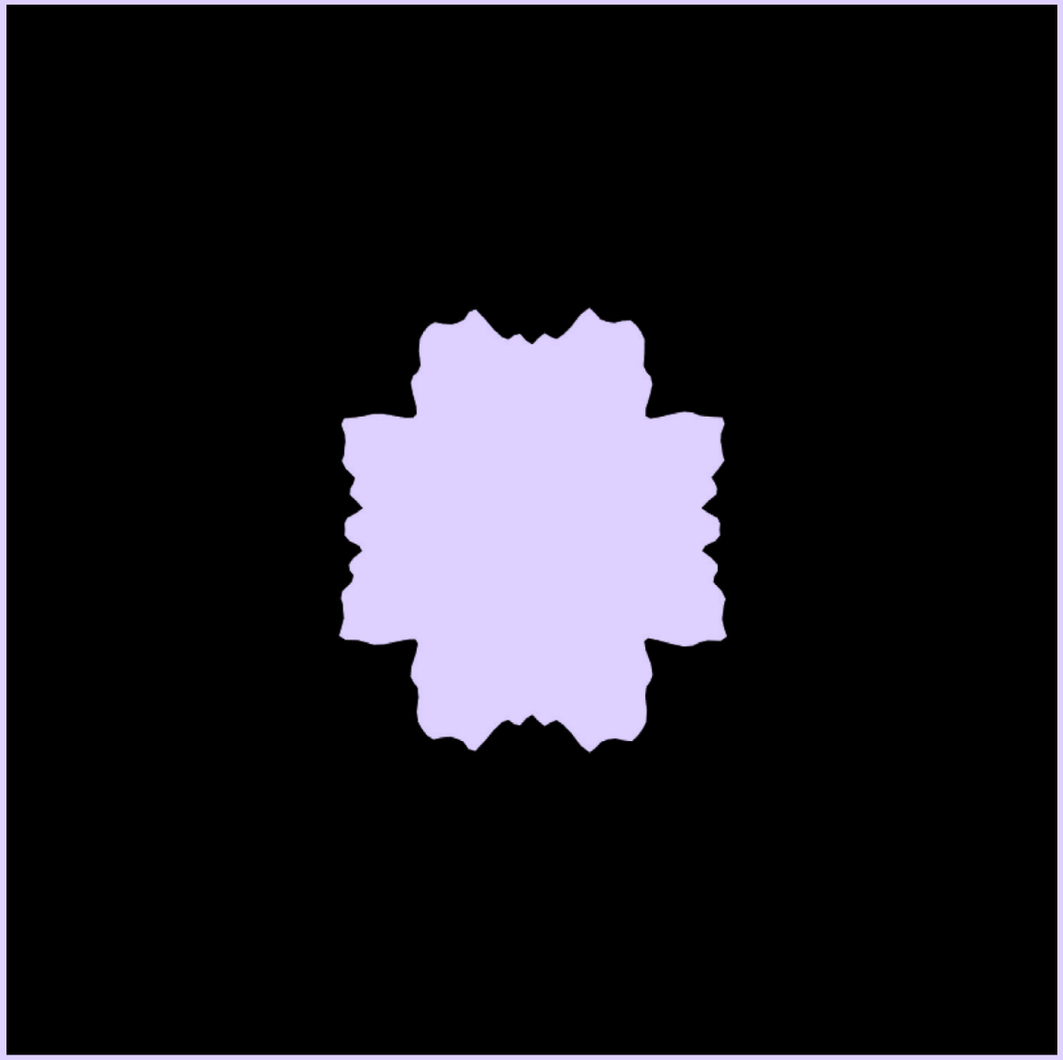
I have contributed to and answered questions on the processing forums, particulary regarding linux and RaspberryPI, where I can offer years of experience. More recently I have been exploring OpenSimplex2 as the default noise implementation for my ruby-processing projects. Another area of interest for me is contextfreeart (a sort of visual sudoku) , and I have developed a JRubyArt DSL, and explored exporting sketches from JRubyArt to render with contextfreeart. I do all of this because I can, and it helps me keep my brain active.











## MICROMUNDOS AMENAZADOS
## THREATENED MICROWORLDS

2013, Medialab-Prado afterARCO
Madrid/ Spain
Photography: Fernanda Ramos
Creative Coder: Nacho Cossío
Made with Processing

The animation was produced with photographies of the inhabitants of the Cabanyal neighborhood in Valencia, who face the threat of losing their homes. In 2010, when the photographs were made, this district, with buildings of the nineteenth century and generations of families living there for more than 300 years, could be bulldozed in order to build a boulevard across it.

In this video, exhibited on Medialab-Prado's digital facade, the portraits turn into a mix of falling faces, representing the unity on their fight.



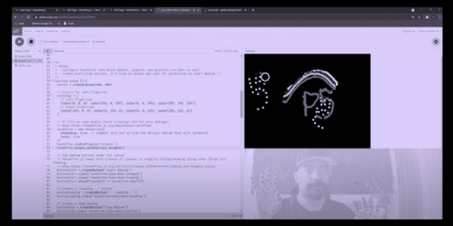
## interacting with p5.js with

# HANDSFREE JS

Duckface Hunt - Face gestures to shoot ducks

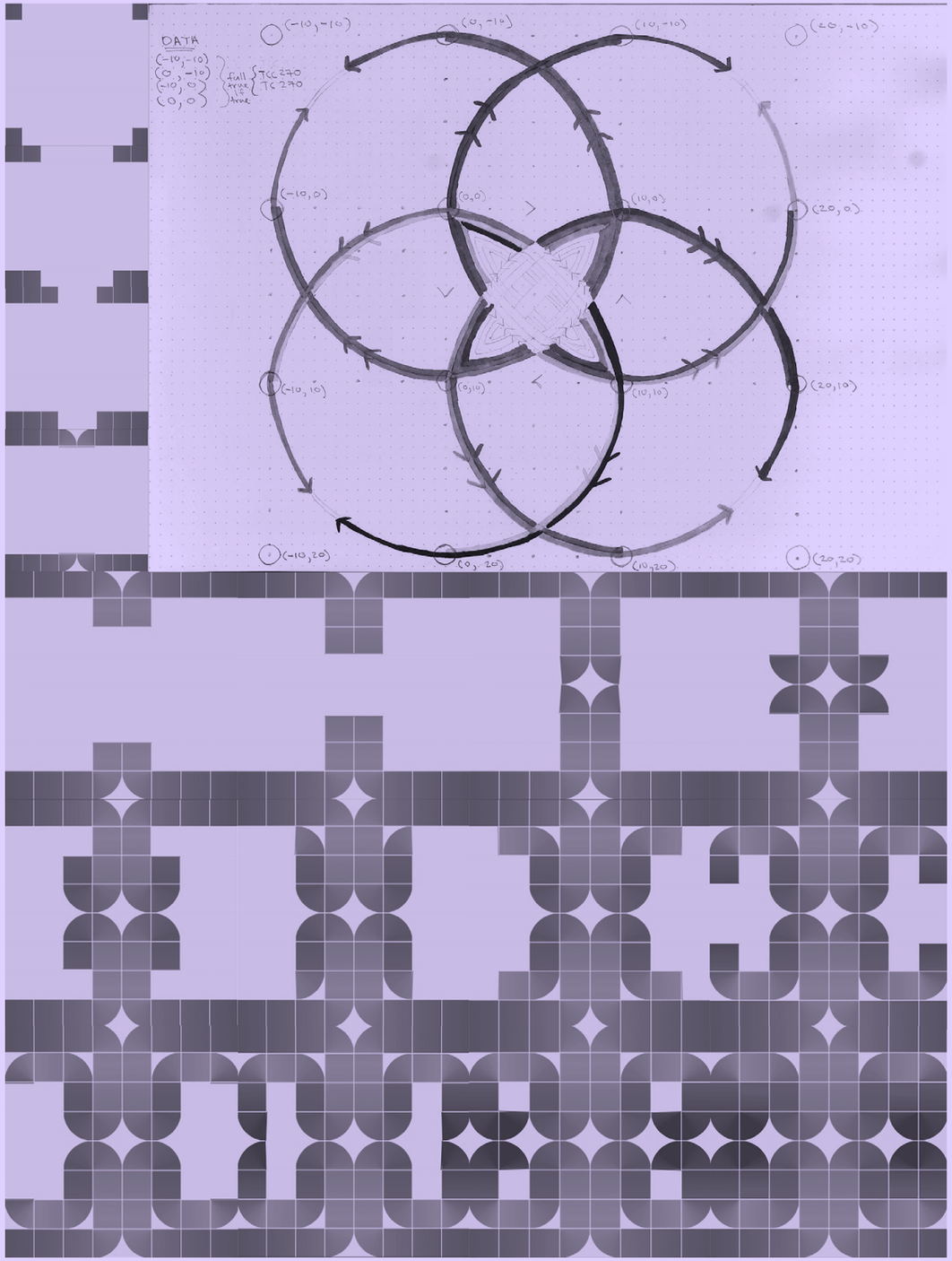

Flappy Pose - Flap arms to fly up and down


Finger painting - Pinch fingers to paint

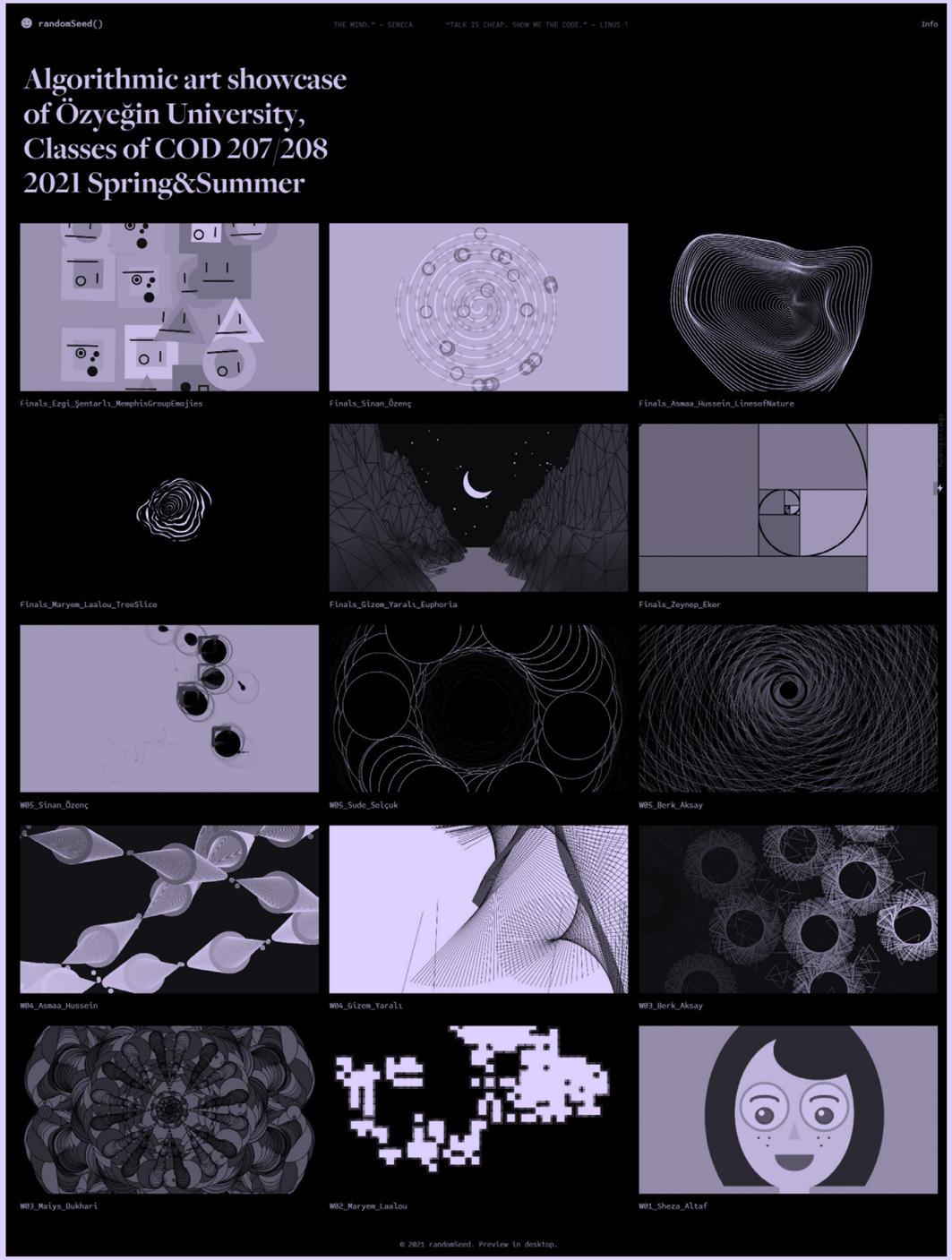


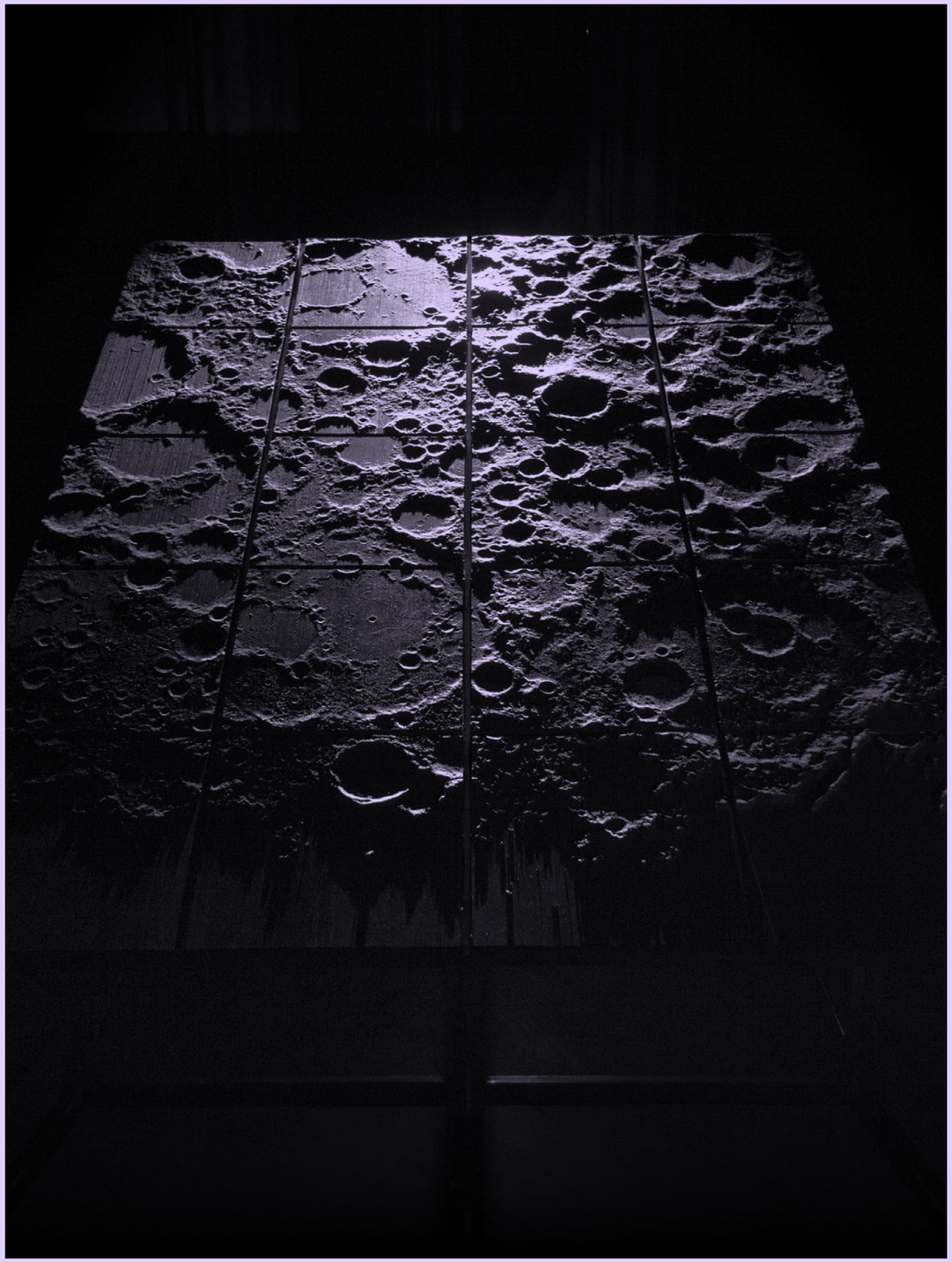
Face Paint - Face gestures to paint on mobile

CodeMedium - Use face gestures to drag code blocks that generate p5.js code hands-free


DATA
(-10,-10)
(0,-10)
(-10,0)
(0,0)
full circle if true
TC≤270
TC≥270
(-10,-10)
(0,-10)
(10,-10)
(20,-10)
(-10,0)
(0,0)
(10,0)
(20,0)
(-10,10)
(0,10)
(10,10)
(20,10)
(-10,20)
(0,20)
(10,20)
(20,20)




randomSeed()
THE MIND." - SENECA
"TALK IS CHEAP. SHOW ME THE CODE." - LINUS T
Info
Algorithmic art showcase of Özyeğin University, Classes of COD 207/208 2021 Spring&Summer
Finals_Ezgi_Şentarlı_MemphisGroupEmojies
Finals_Sinan_Özenç
Finals_Asmaa_Hussein_LinesofNature
Finals_Maryem_Laalou_TreeSlice
Finals_Gizem_Yarali_Euphoria
Finals_Zeynep_Eker
W05_Sinan_Özenç
W05_Sude_Selçuk
W05_Berk_Aksay
W04_Asmaa_Hussein
W04_Gizem_Yarali
W03_Berk_Aksay
W03_Maiys_Bukhari
W02_Maryem_Laalou
W01_Sheza_Altaf
© 2021 randomSeed. Preview in desktop.



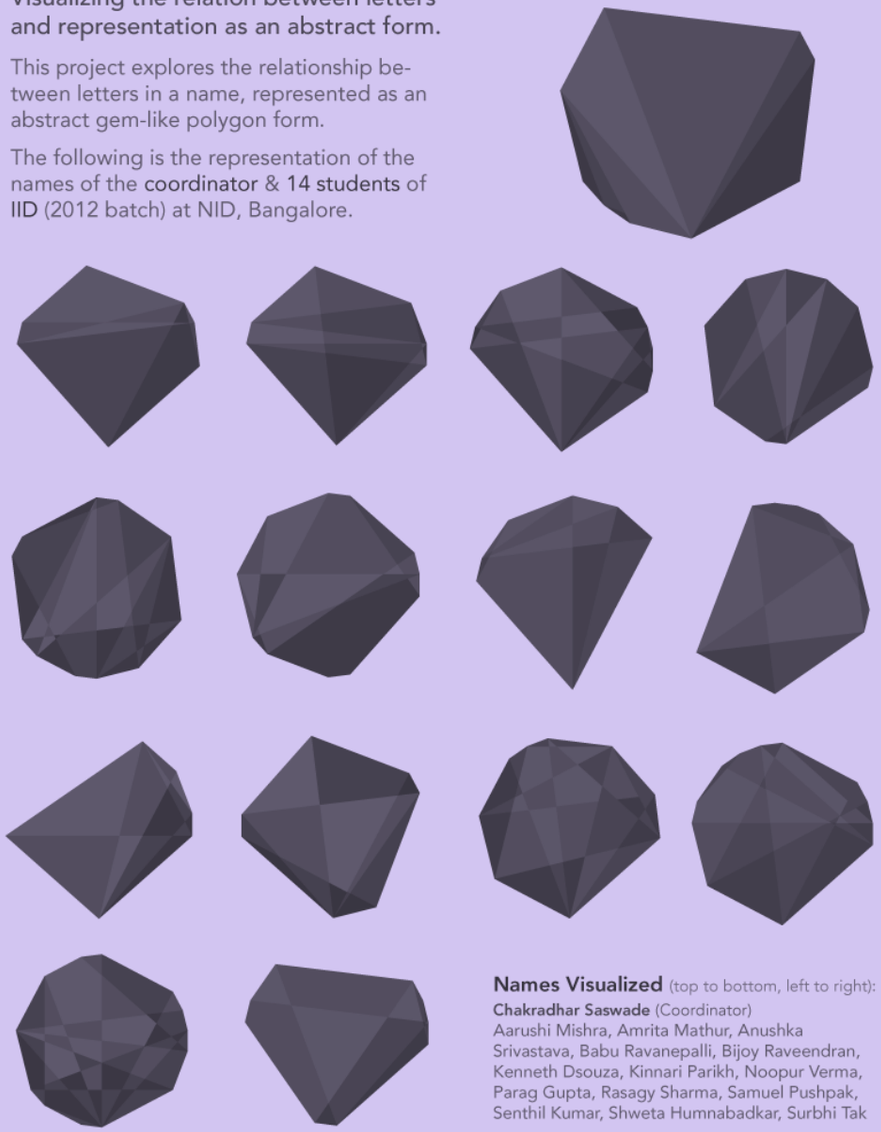


# Gemnam: IID Batch (2012)

**Visualizing the relation between letters and representation as an abstract form.**

This project explores the relationship between letters in a name, represented as an abstract gem-like polygon form.

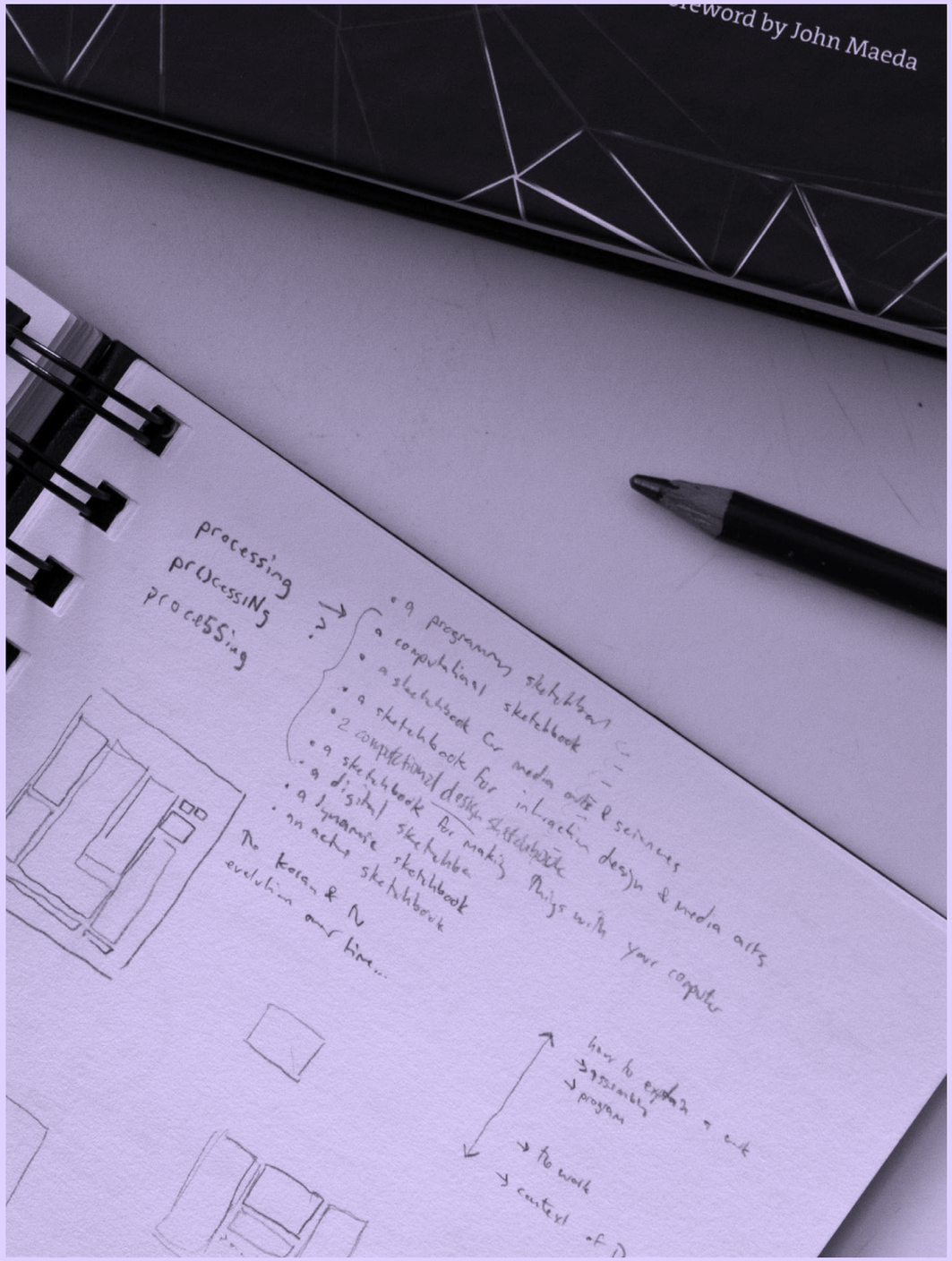
The following is the representation of the names of the coordinator & 14 students of IID (2012 batch) at NID, Bangalore.



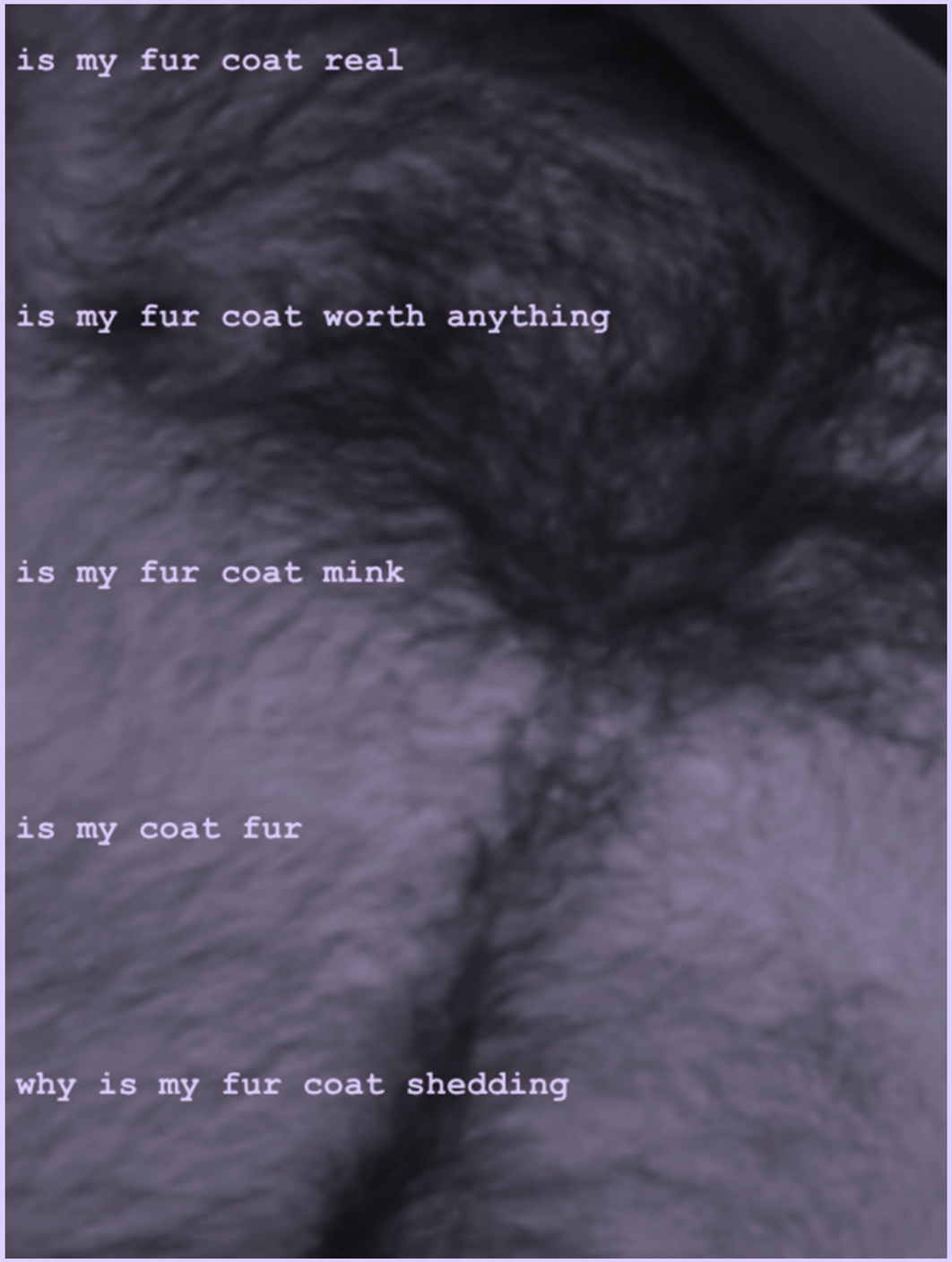
**Names Visualized** (top to bottom, left to right):
**Chakradhar Saswade** (Coordinator)
Aarushi Mishra, Amrita Mathur, Anushka Srivastava, Babu Ravanepalli, Bijoy Raveendran, Kenneth Dsouza, Kinnari Parikh, Noopur Verma, Parag Gupta, Rasagy Sharma, Samuel Pushpak, Senthil Kumar, Shweta Humnabadkar, Surbhi Tak

Data Visualization Exploration by **Rasagy Sharma** (IID)

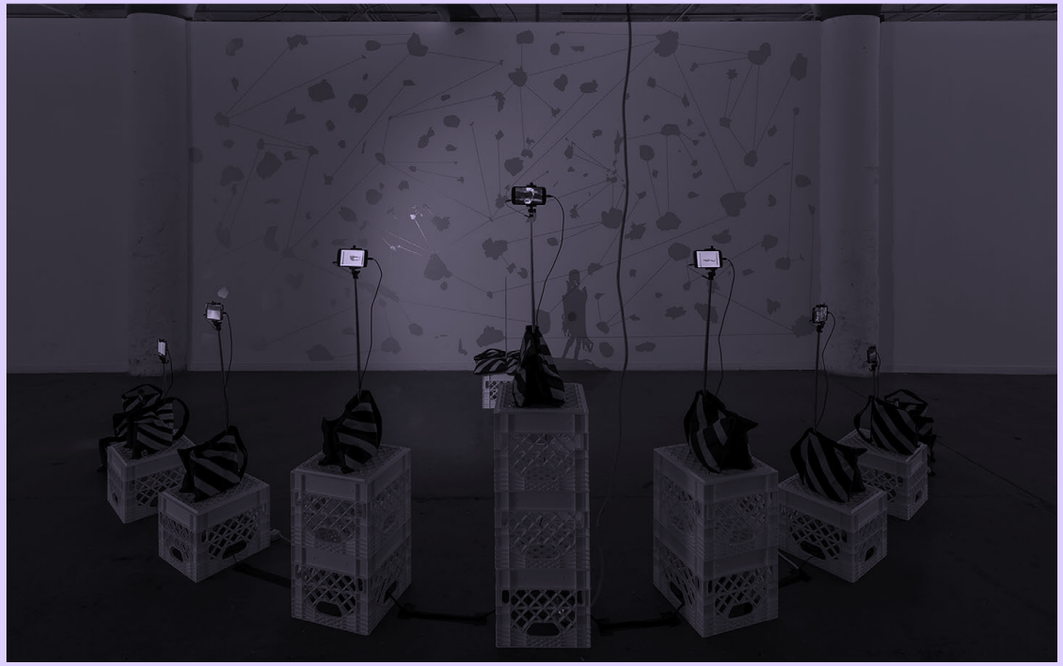
FABIN RASHEED | WWW.NURECAS.COM



@REAS


is my fur coat real
is my fur coat worth anything
is my fur coat mink
is my coat fur
why is my fur coat shedding

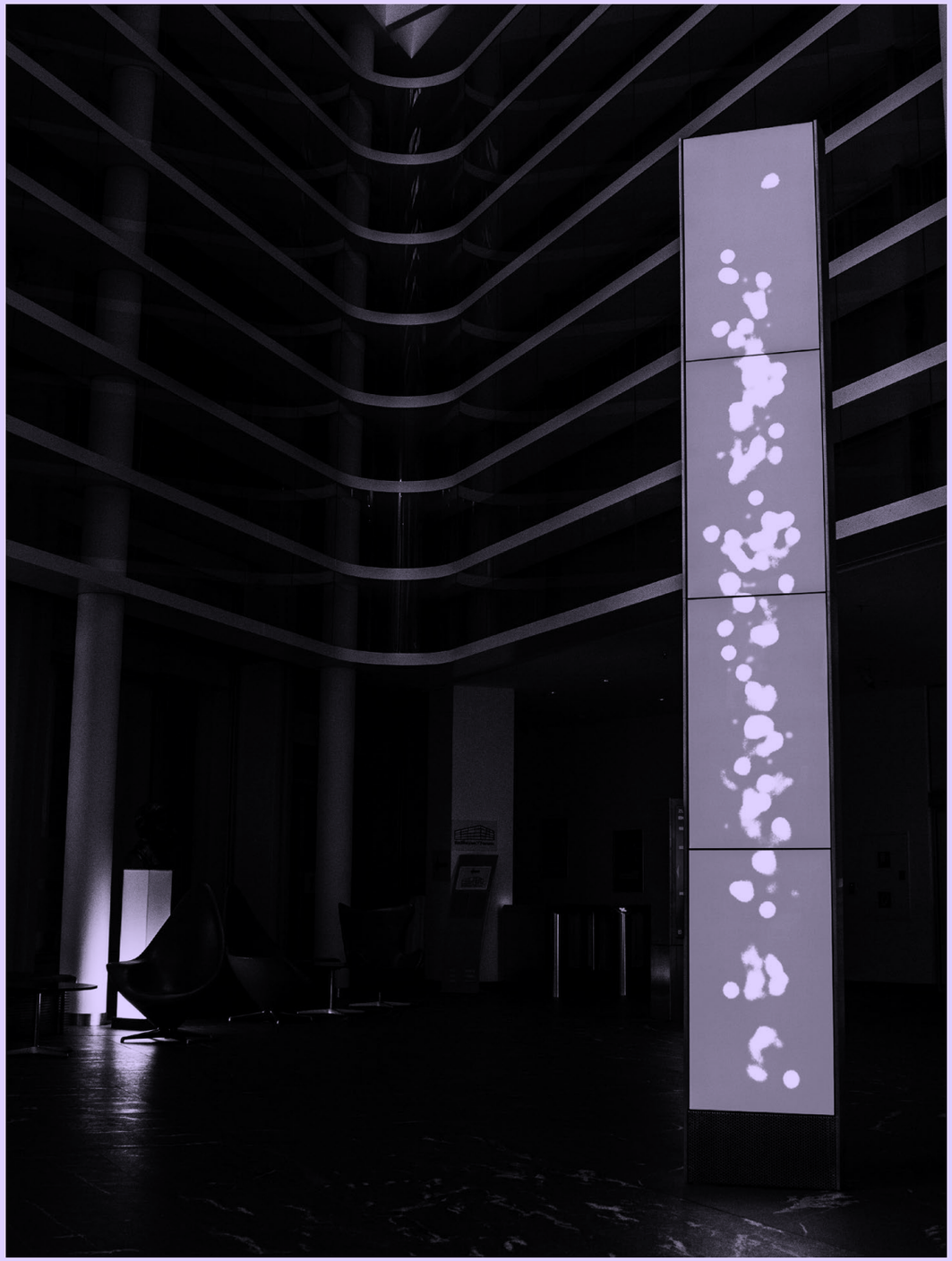


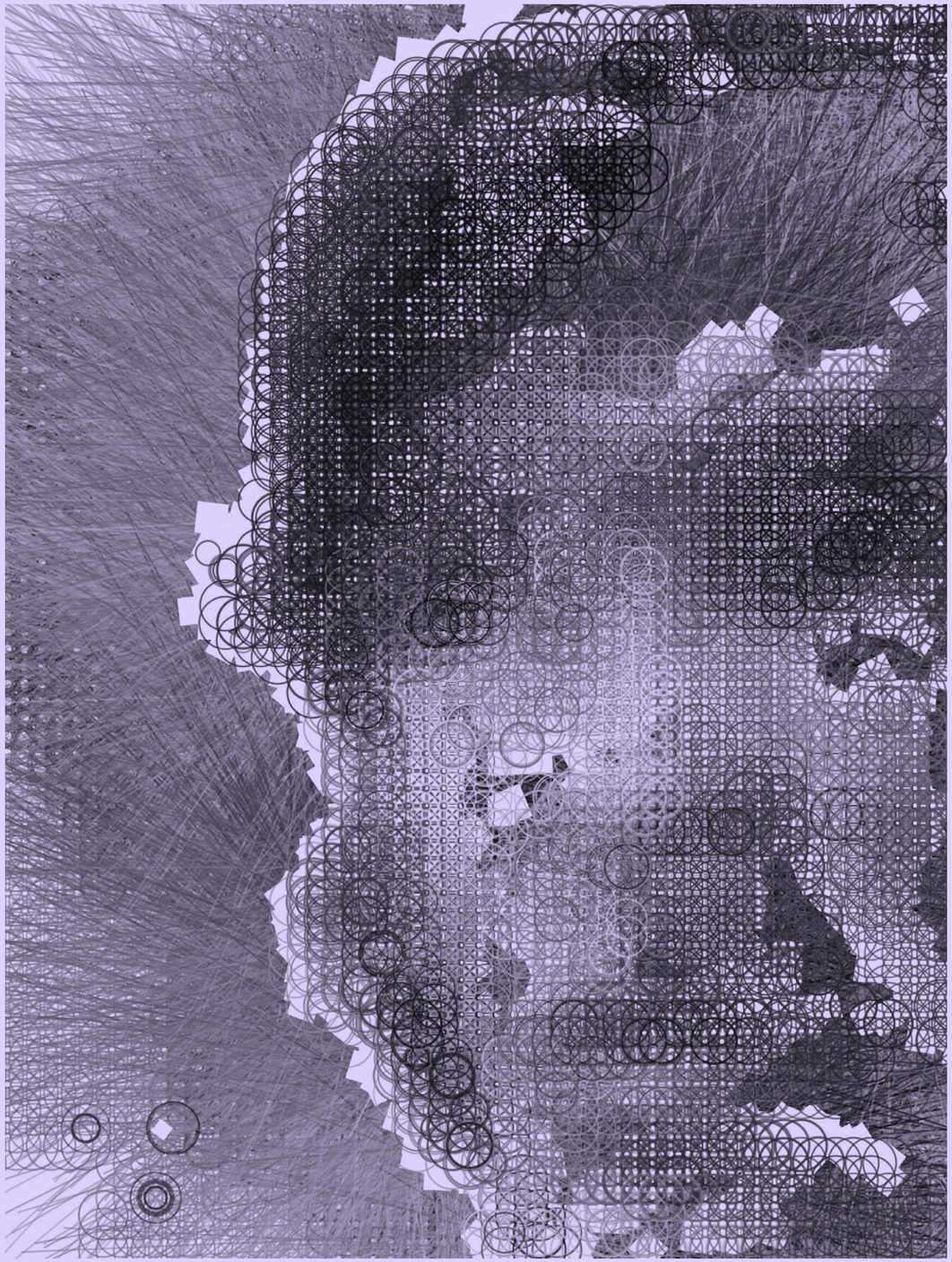


# #つぶやきProcessing
# Tattoo Sticker

@reona396

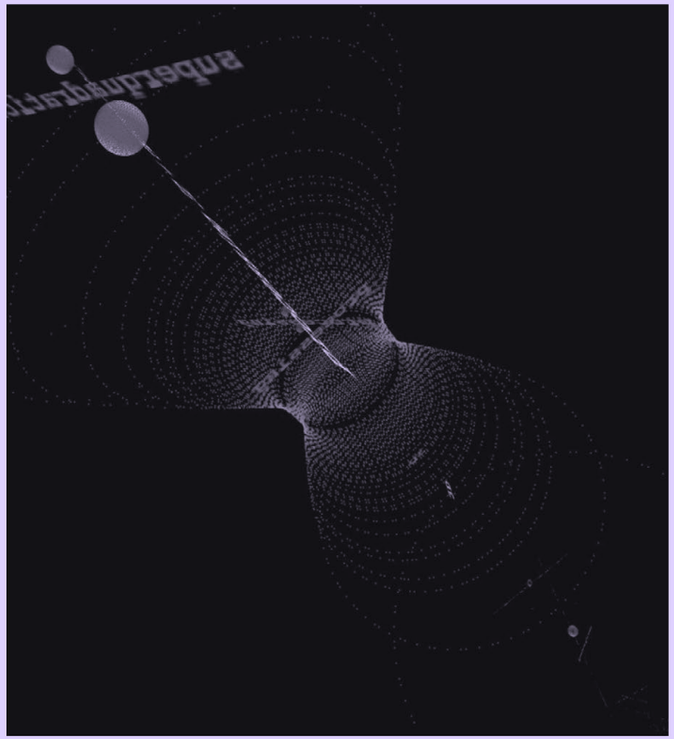


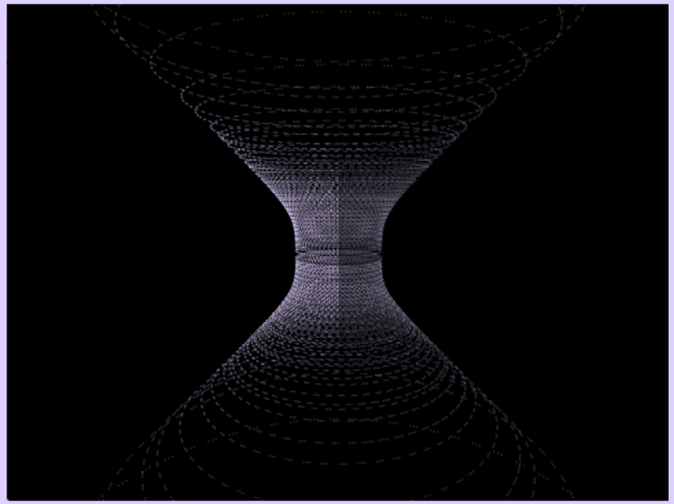






# Messages vers le futur





**Messages vers le futur** was a public installation designed for the event La Nuit de Chercheurs 2012 in France. Participants logged into their Twitter account and wrote a message with the hashtag #messagesFutur. The text of the message appeared in a large-size projection. Each message was also represented as a 3D sphere that travelled through a wormhole and then disappeared in the distance.

The wormhole was indeed a hyperbolic mathematical figure coded in Processing. Tweets however got stored in a Gmail account and will be sent back to their users this year, later in 2022.

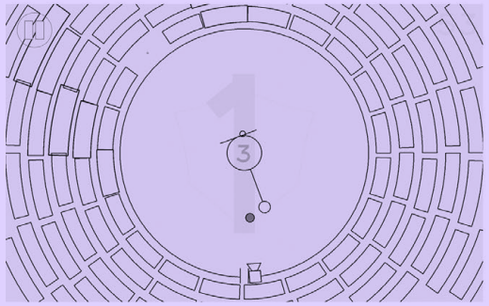
Concept and development:
**Everardo Reyes**
Thanks to **Eleanor Dare & Kostas Terzidis** for their guidance
Paris, France, 2012

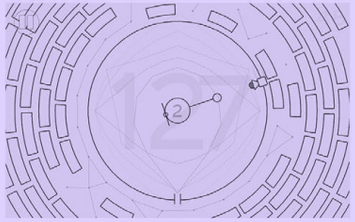
rich.gg

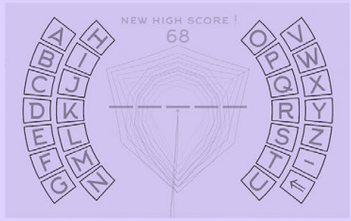
organized PCD Reunion Island 2019

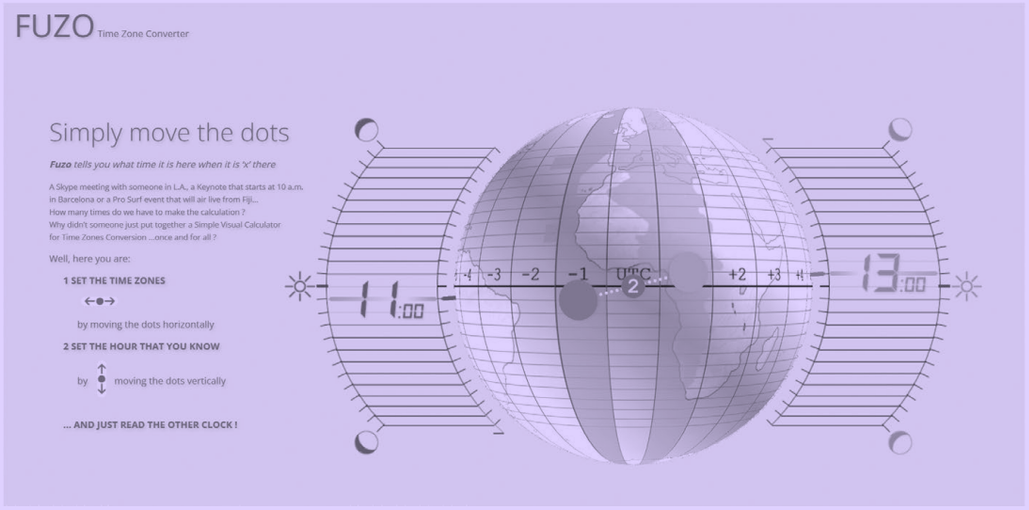
**Digital Architect**,
Rich started with VRML back in 1998
and is so thankful for **Processing**
to welcome beginners and artists
in the world of code

You can find his **Processing Android**
creations on the Play Store
and his full **p5.js website** at rich.gg

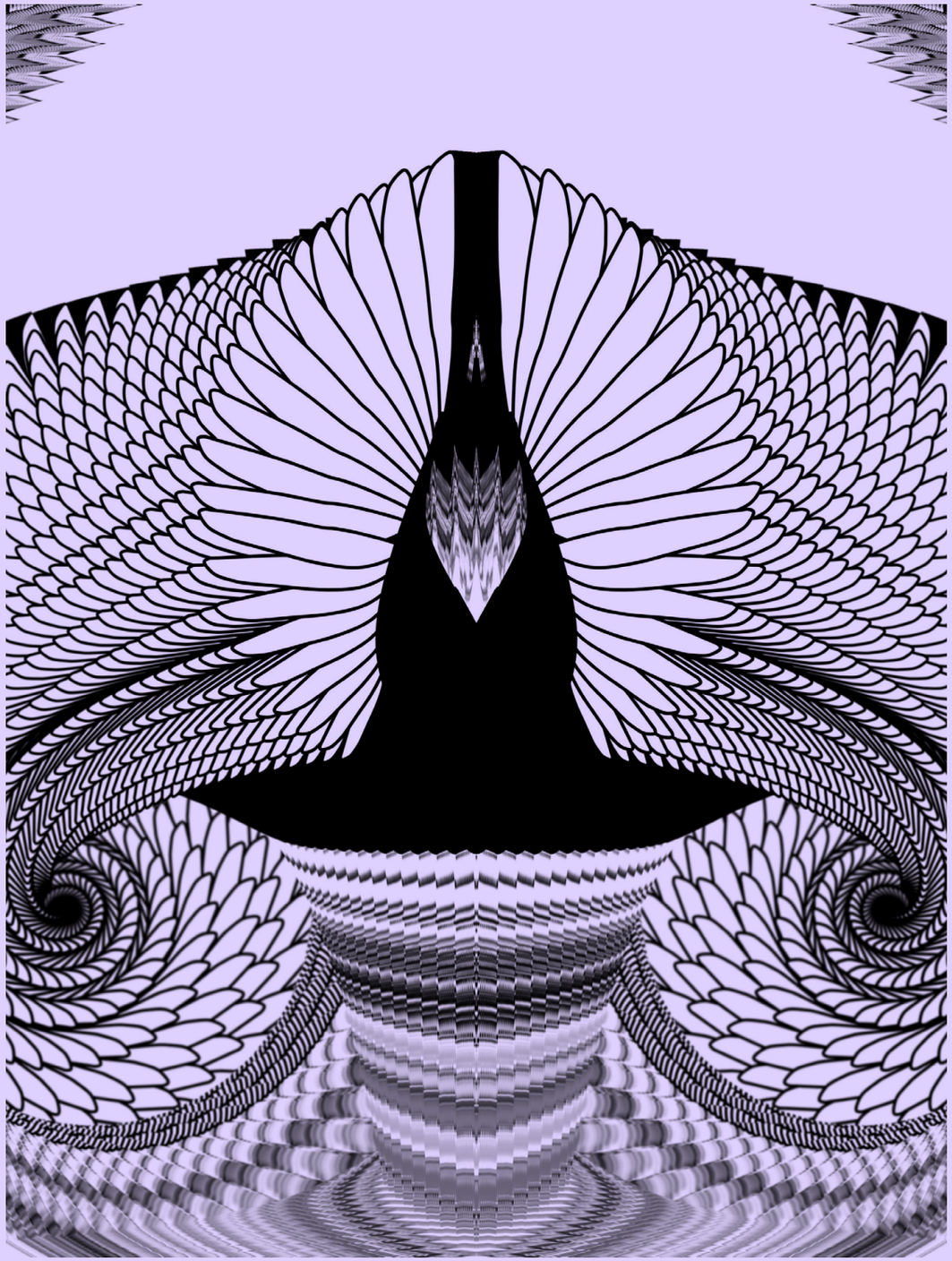


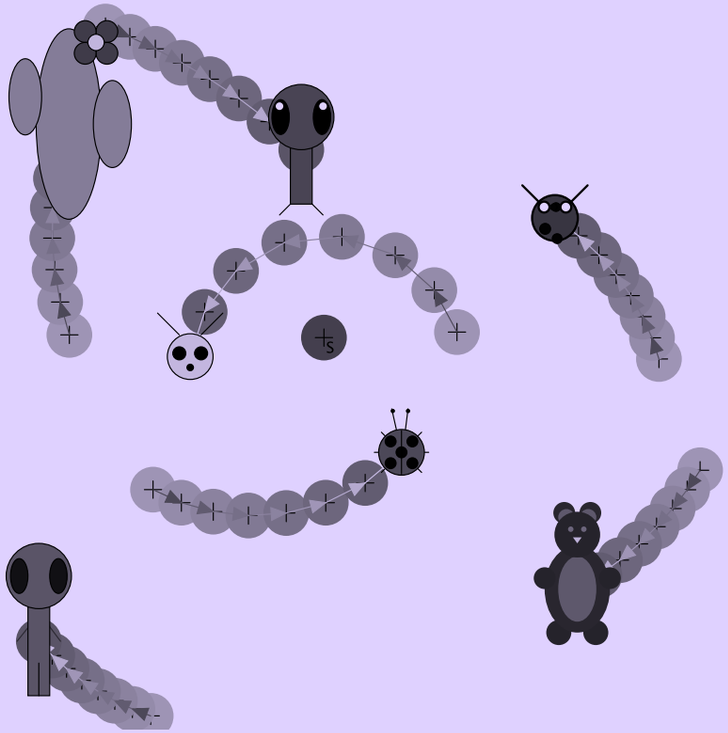









Several objects on their orbit through the Processing.py universe

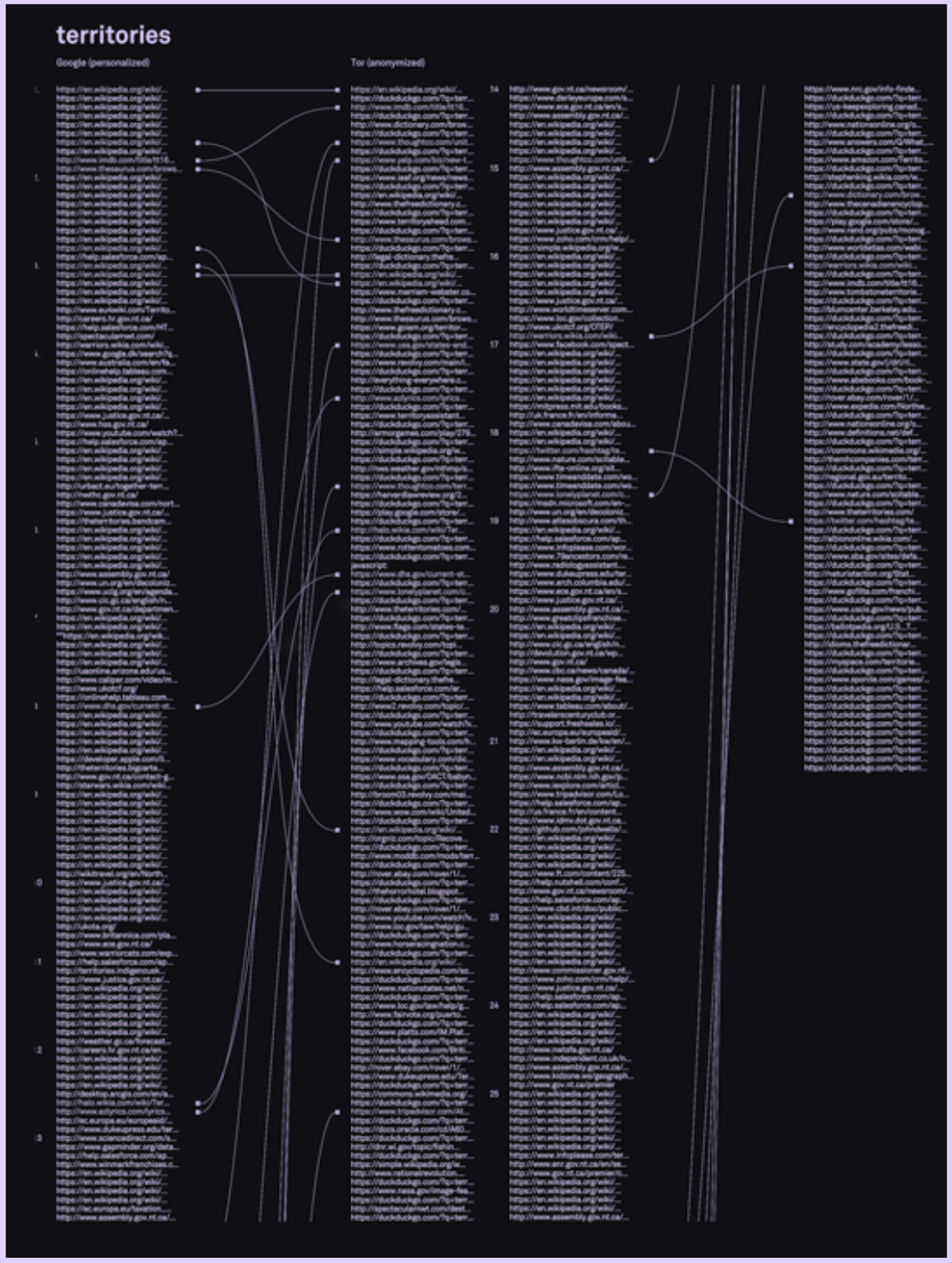
This summer term, I taught the topic “vectors” in two ways: first, the mathematics of two-dimensional vectors, as usual; second, a short programming course in Python, followed by weekly assignments in Processing.py.

Each of my students at Khevenhüller Gymnasium (Linz, AT) created their own object; had it move across the canvas; added force; learnt about speed and acceleration, and, above all, about vectors.

In the last assignment, we sent all these objects – aliens, ladybirds, cacti, teddy bears, and the like – into orbit. There they will continue to float as long as Processing’s Python mode works … thanks to all who keep this universe running!

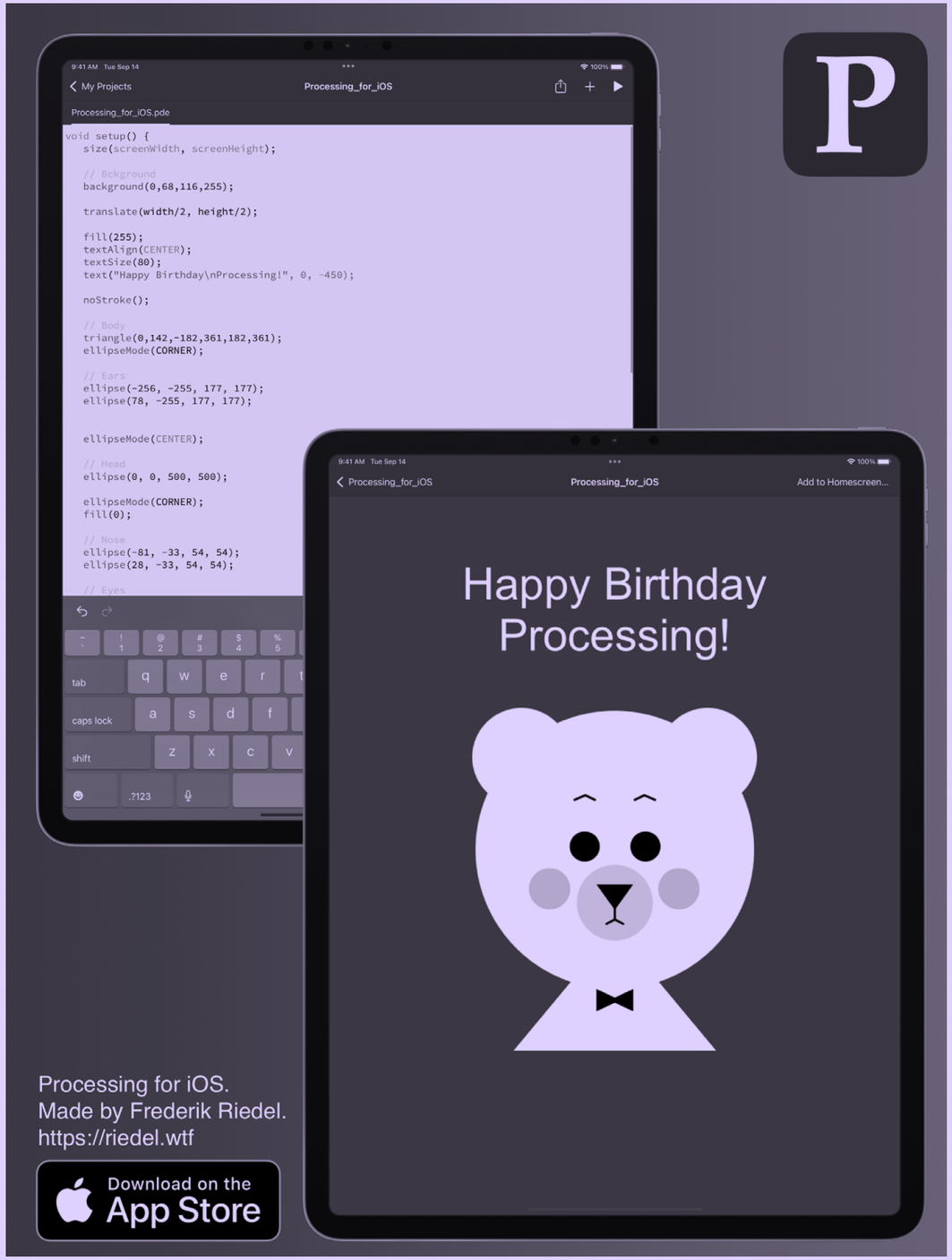
*Flying objects were created by Elisa Antelmann, Paula Maria Bauer, Mithad Jan Bogner, Luisa Mayr, Sina Mayr, Anton Pargfrieder, and Elisabeth Schimana; additional code and documentation by Roland Richter. The code is available under a GPL licence at* `https://github.com/rric/processing-vector`

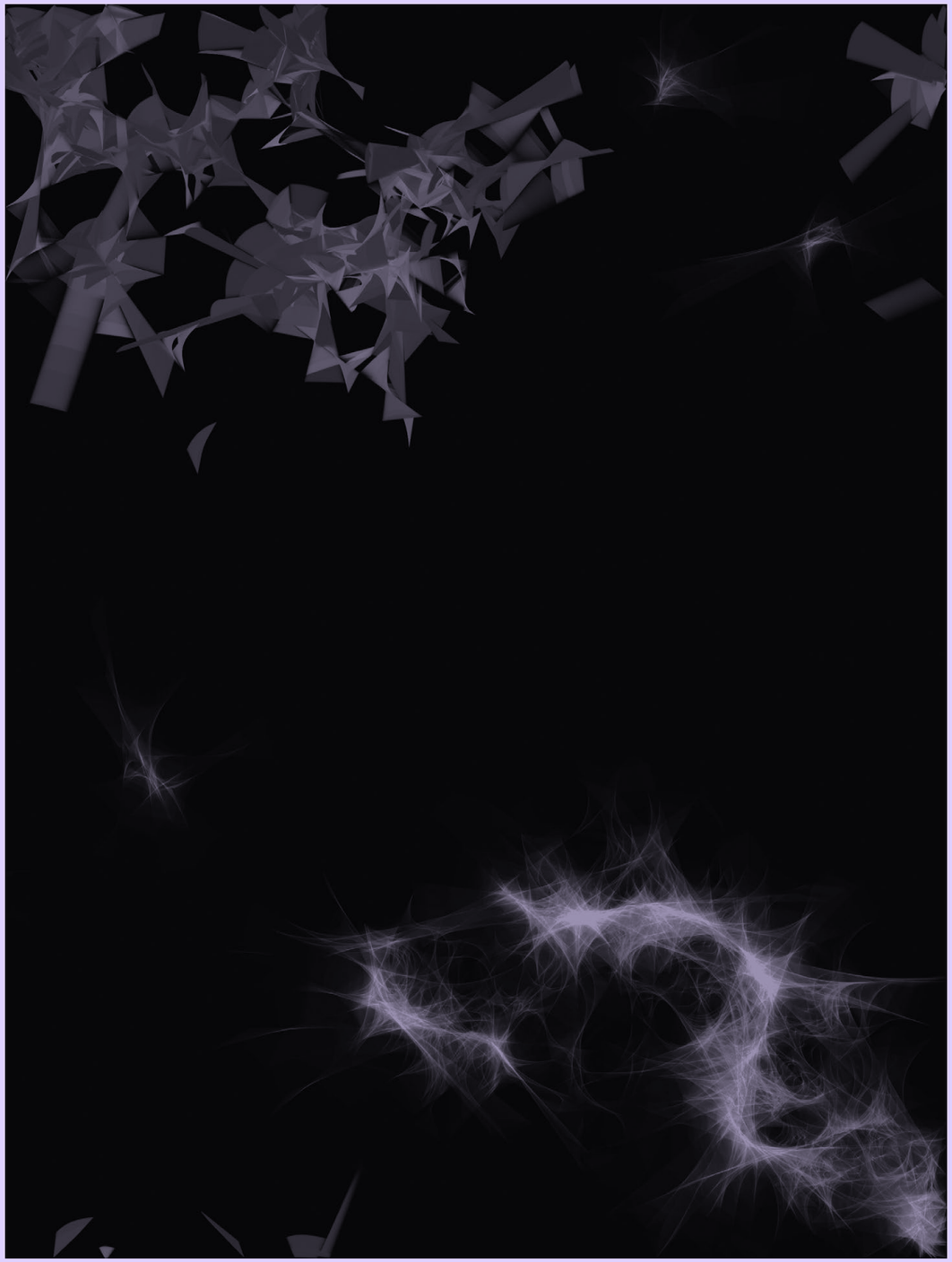





Processing, we have spent many hours in the same room..
We've argued, wrestled, and laughed together.
Chris Ried - @generatecoll / #genartclub
MAY IT NEVER STOP! Happy 20th Birthday!

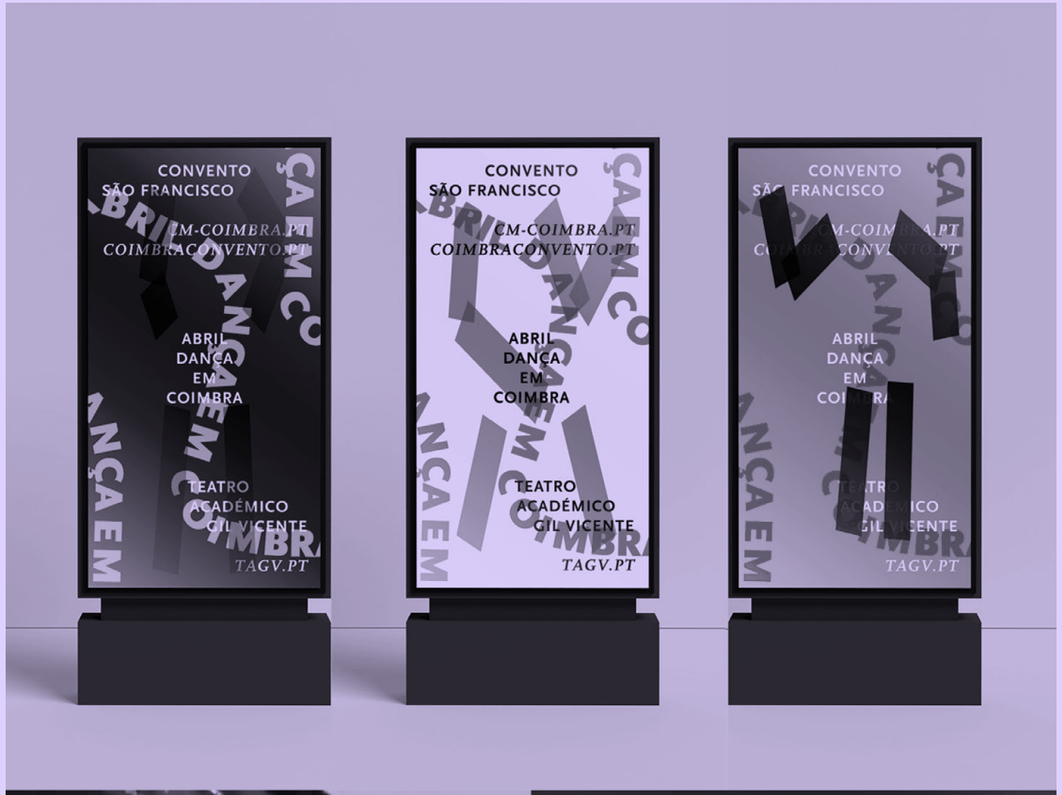



My Projects
Processing_for_iOS
Processing_for_iOS.pde
Add to Homescreen...
Happy Birthday
Processing!
Processing for iOS.
Made by Frederik Riedel.
https://riedel.wtf
Download on the
App Store






**Logika / Panna**
10” Vinyl
Cover art made in Processing
Record Store Day Japan 2018




/CÁTIA ROÇA



# PCD 2021

**PROCESSING COMMUNITY DAY 2021**

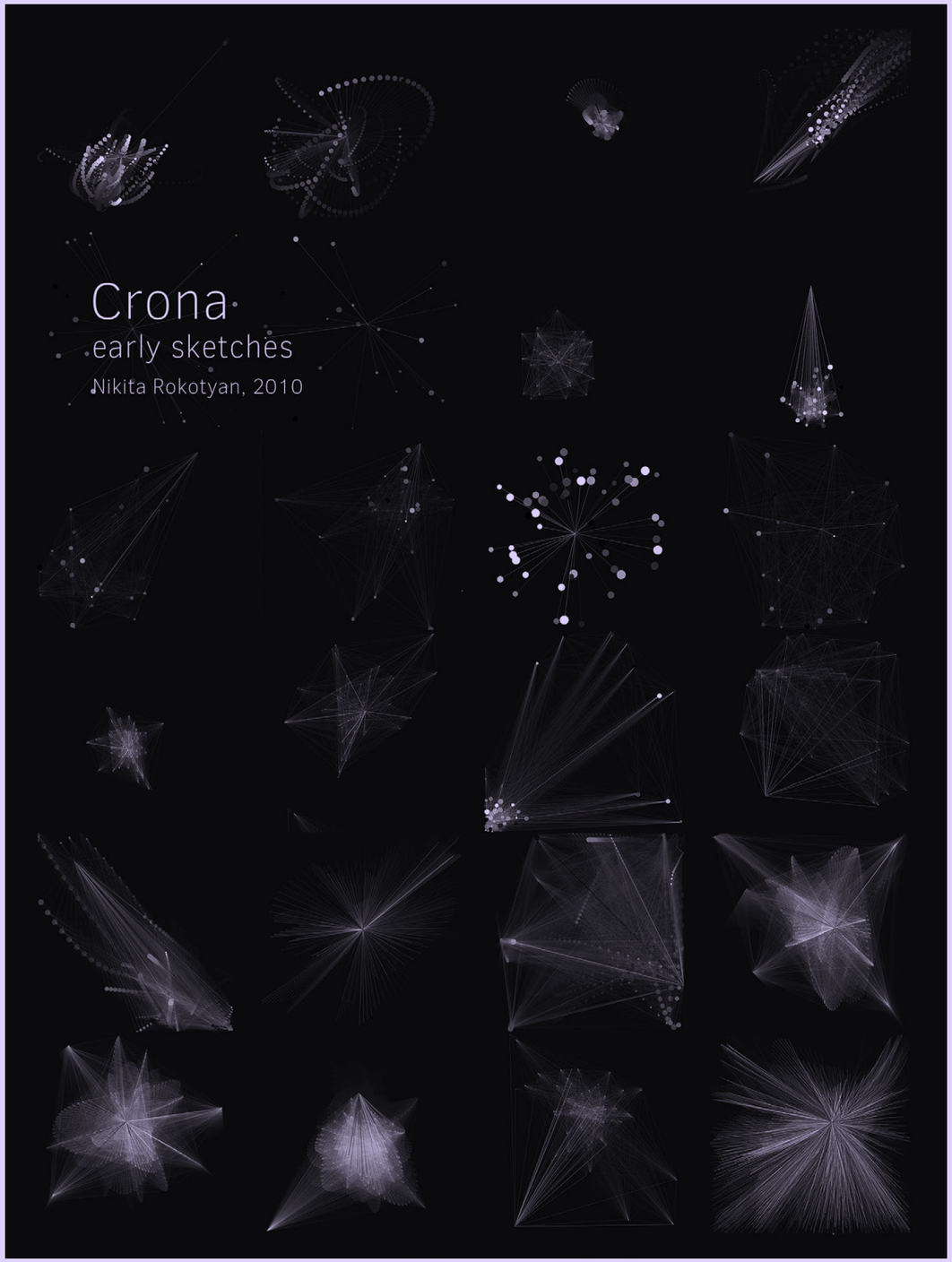
Celebrating 20 years of Processing with talks and contributions by

Patrik Hübner
Vera van de Seyp
Dr. Martin Lorenz
Casey Reas
Sander Sturing

all talks are now online!
timrodenbroeker.de/pcd2021

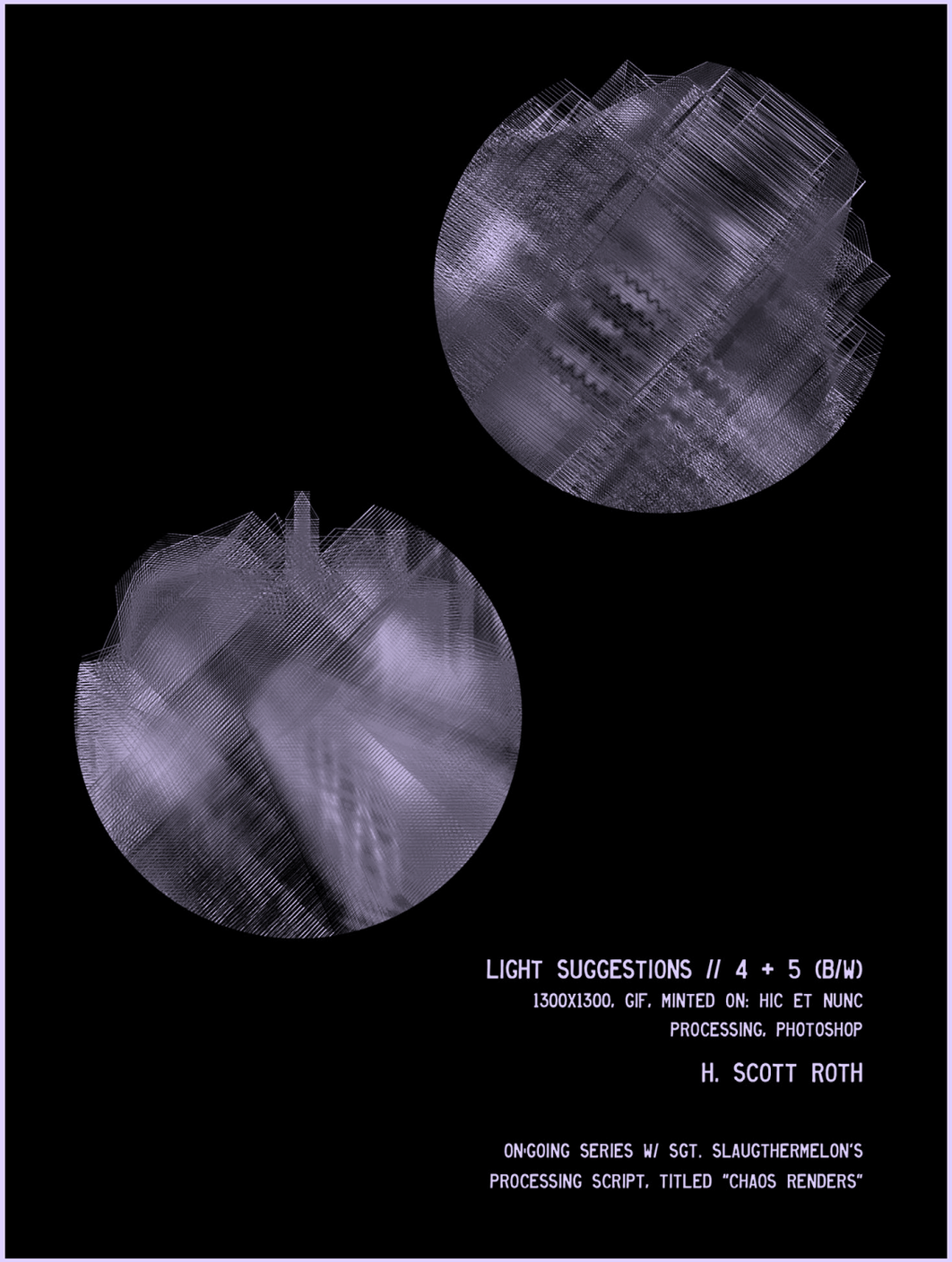
trcc

TIM RODENBRÖKER, TIMRODENBROEKER.DE


Crona
early sketches
Nikita Rokotyan, 2010




RED SCARE
SINOPHOBIA
press keys: a, s, d, f, g

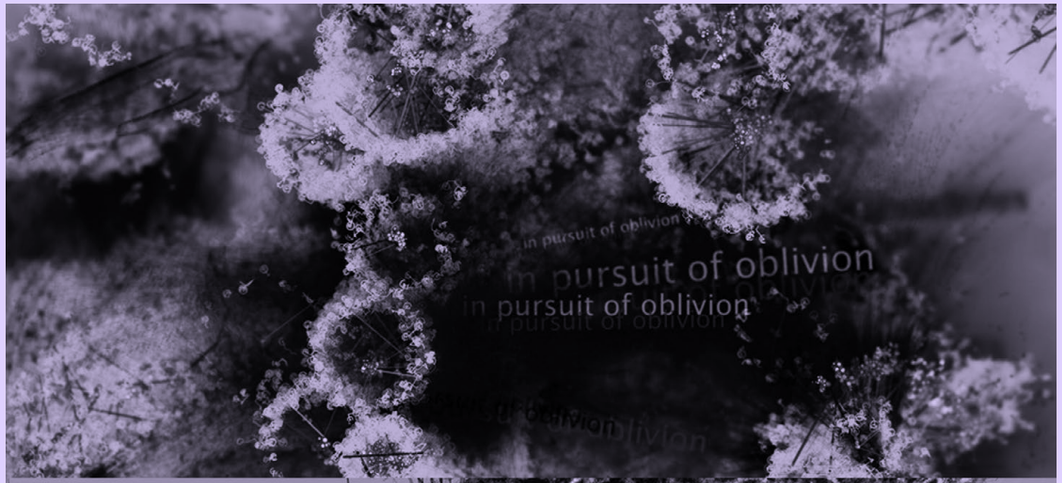


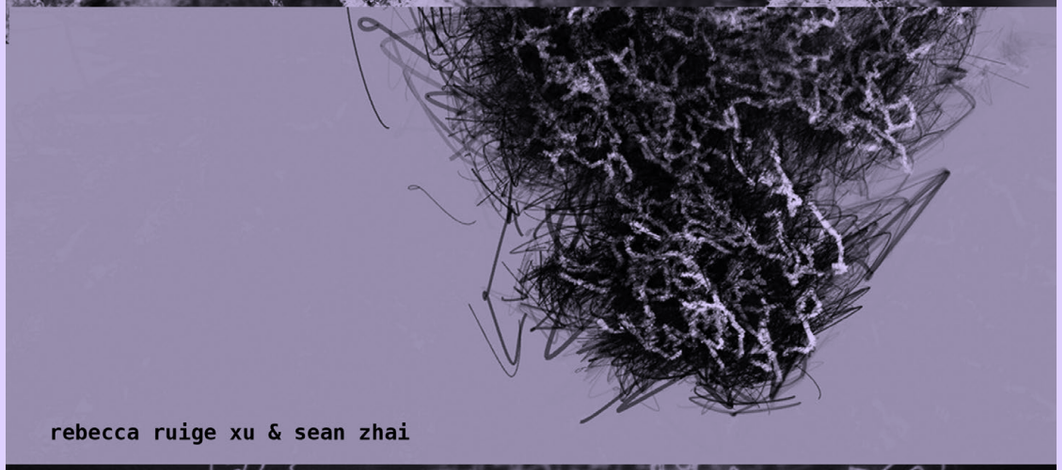
LIGHT SUGGESTIONS // 4 + 5 (B/W)
1300X1300. GIF. MINTED ON: HIC ET NUNC
PROCESSING. PHOTOSHOP

H. SCOTT ROTH

ON-GOING SERIES W/ SGT. SLAUGTHERMELON'S
PROCESSING SCRIPT. TITLED "CHAOS RENDERS"

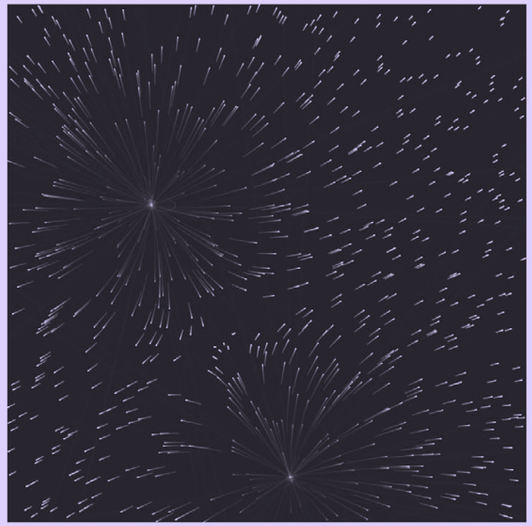


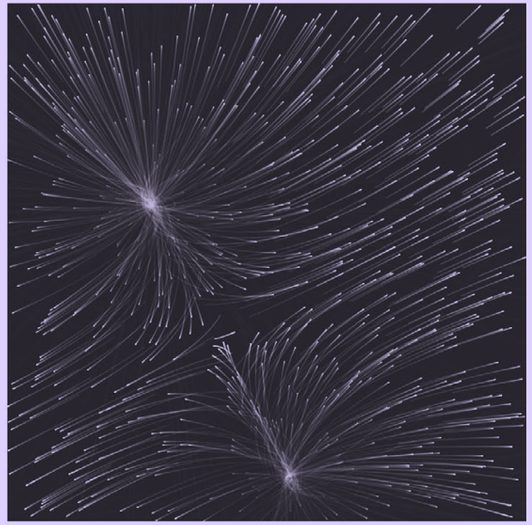


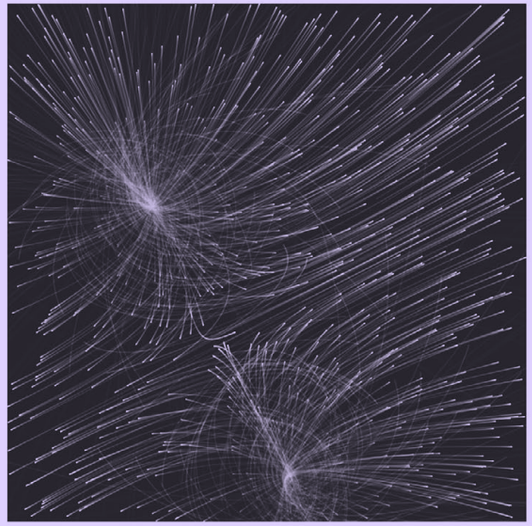







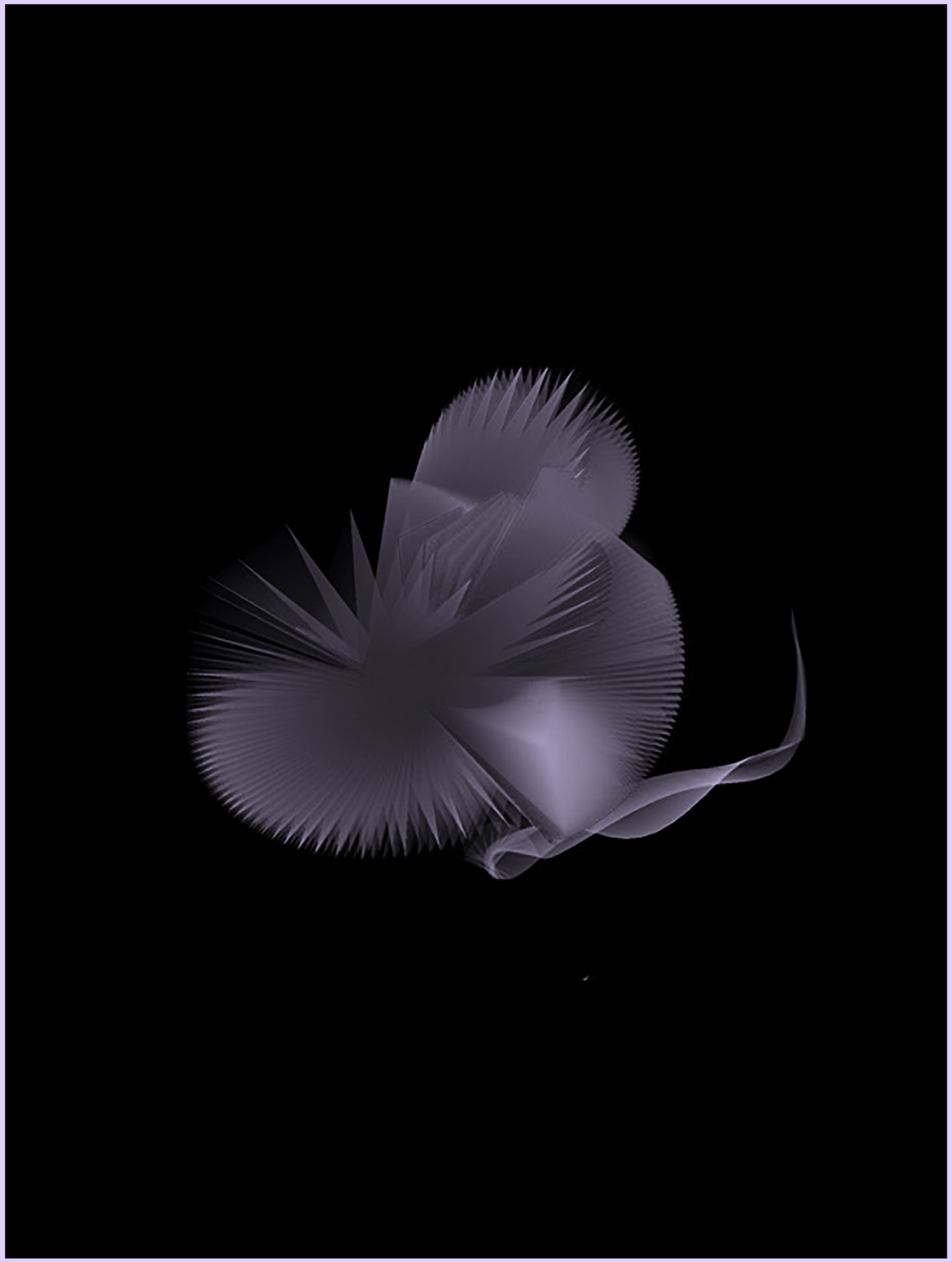






**001**

1583483718263 1583483731105
1583483746819



SOUNDBIRD (2008) BY RUX (RUI PEREIRA) CON 467 782


RVIG, GENERATIVE ARTIST, @RVIG_ART CON 468 783


Tom-Lucas Säger

Processing Cake

Processing turned 20, Hooray!

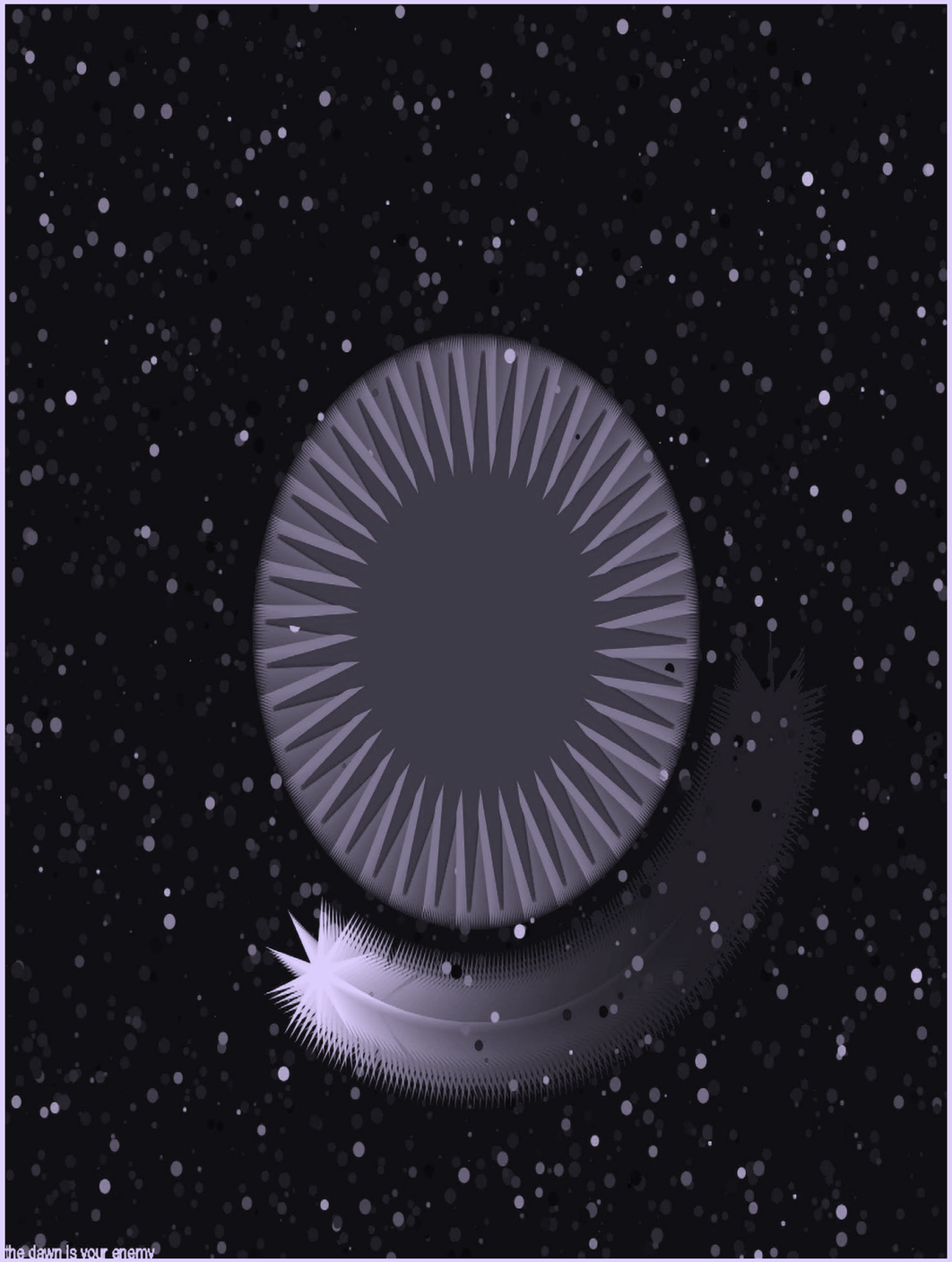
This means you need to eat a lot of cake.

Scan the QR-Code and enjoy a delicious piece of digital cake, but avoid those bad veggies!

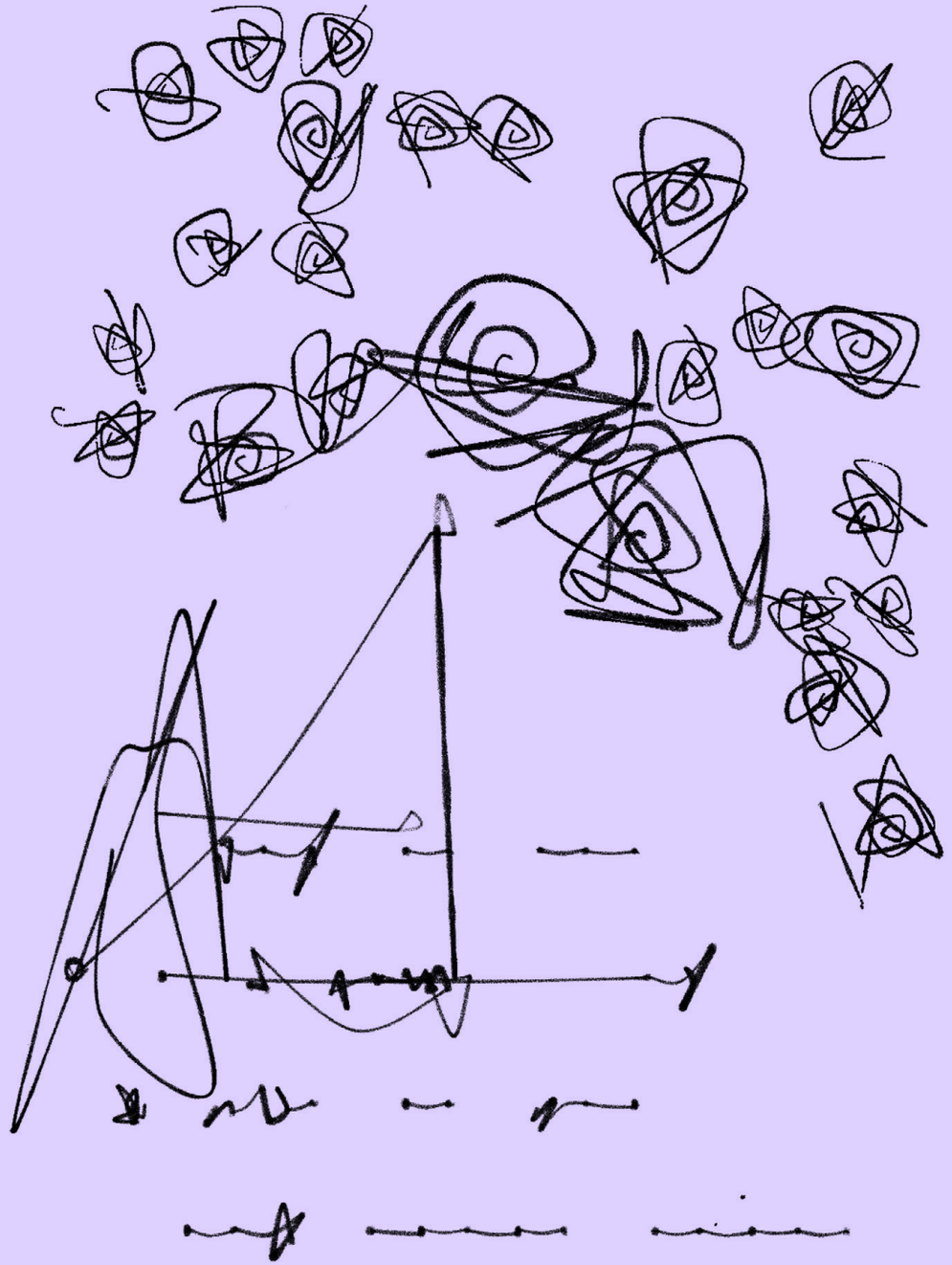
https://bit.ly/Processing_Cake

Made with p5.js and ml5.js by Tom-Lucas Säger, Creative Technologist from Hamburg and proud member of the Processing Community.

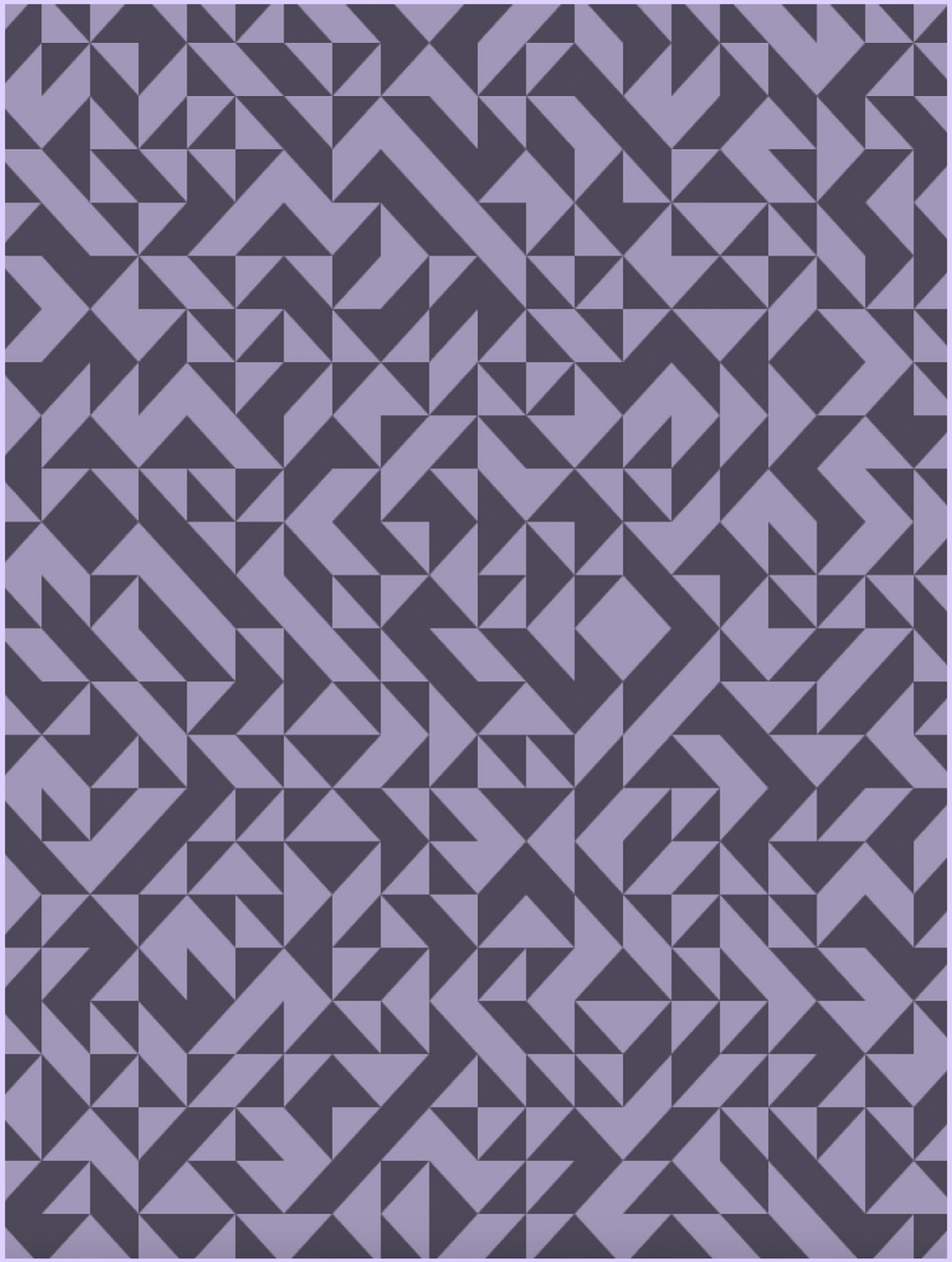
Instagram: @tlsaeger
Twitter: @t00may
Web: tlsaeger.de





Processing :: a love letter
Sarah Ridgley | @sarah_ridgley

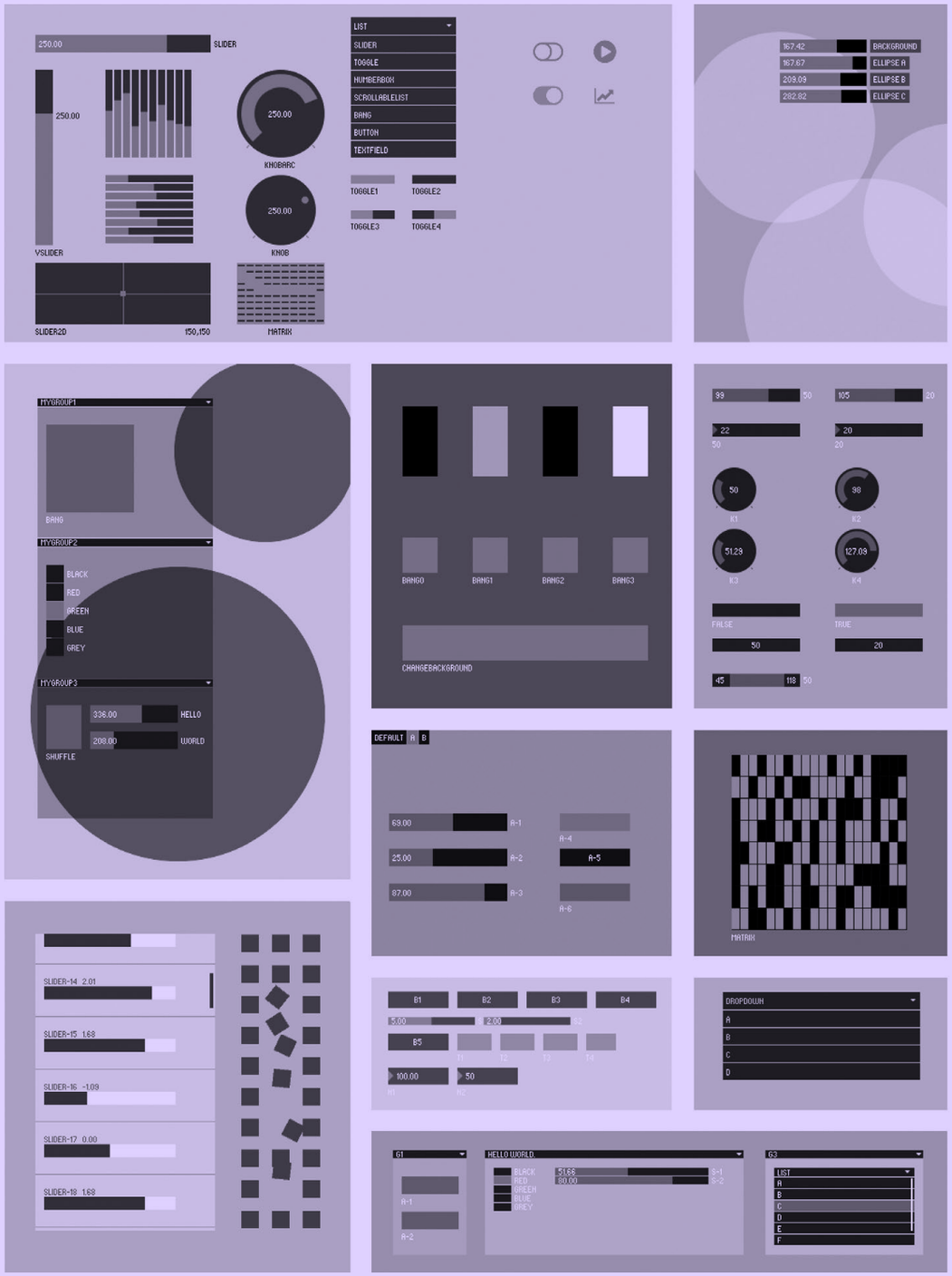


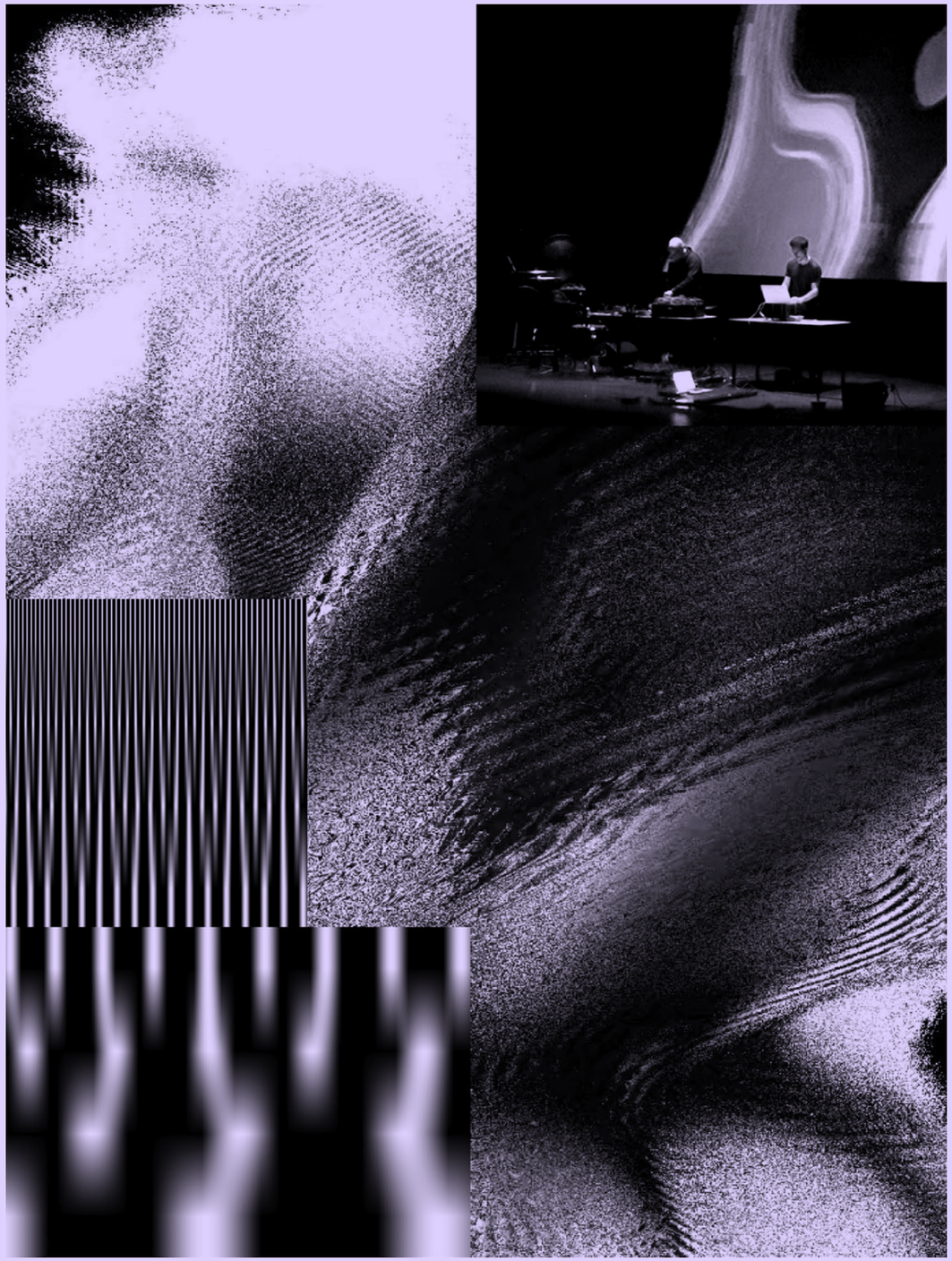


Processing makes you a magician.
Anyone can create magical moments in this
digital world. It reminded me one of artistic trends
called Magical Realism, where fantastical elements
are considered as something normal in
the real world. When I explore Twitter
where I condier it is a real world, I
saw magical art works created by Processing.
Just like magic, there are small tricks
to make it happen. If you know a bit of
coding, you can be a magician immediately.
Also, if you master the rules of Processing, only
your creativity is the limit. Since I happend to
know this community, I have met a lot of
illusionists. It has been hour to
be part of this community
and have fun at
making magics.
What does being
a part of the Processing
community mean
to me? Processing
gives me a place
to be myself and
makes me feel that
I can decide what
I do



```
float edge = 120;
float d = sqrt(edge*edge*2);
float d2 = d/2;
float rot = 0;

void setup() {
  size(2250, 3000);
  smooth(8);
  fill(0, 40);
  noStroke();
}

void draw() {
  background(255);
  float x1 = cos(rot) * d2;
  float y1 = sin(rot) * d2;
  float x2, y2, r;
  for (float x=0; x<=width + d; x += d) {
    for (float y=0; y<=height+d; y += d) {
      r = x*rot/12000;
      x1 = cos(r) * d2 ;
      y1 = sin(r) * d2;
      beginShape();
      vertex(x + x1, y + y1);
      vertex(x - y1, y + x1);
      vertex(x - x1, y - y1);
      vertex(x + y1, y-x1);
      endShape(CLOSE);
      x2 = x + d2;
      y2 = y + d2;
      beginShape();
      vertex(x2 + x1, y2 + y1);
      vertex(x2 - y1, y2 + x1);
      vertex(x2 - x1, y2 - y1);
      vertex(x2 + y1, y2 - x1);
      endShape(CLOSE);
    }
  }
}
```


250.00
SLIDER
250.00
KNOBARC
VSLIDER
250.00
KNOB
SLIDER2D
150,150
MATRIX
LIST
SLIDER
TOGGLE
NUMBERBOX
SCROLLABLELIST
BANG
BUTTON
TEXTFIELD
TOGGLE1
TOGGLE2
TOGGLE3
TOGGLE4
167.42
BACKGROUND
167.67
ELLIPSE A
209.09
ELLIPSE B
282.82
ELLIPSE C
MYGROUP1
BANG
MYGROUP2
BLACK
RED
GREEN
BLUE
GREY
MYGROUP3
336.00
HELLO
208.00
WORLD
SHUFFLE
BANG0
BANG1
BANG2
BANG3
CHANGEBACKGROUND
99
50
105
20
22
50
20
20
50
K1
98
K2
51.29
K3
127.08
K4
FALSE
TRUE
50
20
45
118
50
DEFAULT
A
B
69.00
A-1
25.00
A-2
87.00
A-3
A-4
A-5
A-6
MATRIX
SLIDER-14 2.01
SLIDER-15 1.68
SLIDER-16 -1.09
SLIDER-17 0.00
SLIDER-18 1.68
B1
B2
B3
B4
5.00
2.00
85
F1
F2
F3
F4
100.00
50
M1
M2
DROPDOWN
A
B
C
D
61
A-1
A-2
HELLO WORLD.
BLACK
RED
GREEN
BLUE
GREY
51.66
80.00
B-1
B-2
63
LIST
A
B
C
D
E
F



jupyter py5 notebook

Logout

File Edit View Insert Cell Kernel Widgets Help py5

Run Markdown git nbdiff

# Welcome to py5!

## py5 is a new version of Processing for Python 3.8+

The goal of py5 is to create a version of Processing that is integrated into the Python ecosystem. Built into the library are thoughtful choices about how to best get py5 to work with other well known Python libraries and tools. This includes two custom Jupyter Notebook kernels, one of which was used to create the notebook you are reading right now.

In [1]:
```
import numpy as np
```

Creative coders can build a py5 Sketch by writing `setup()` and `draw()` functions.

In [2]:
```
def setup():
    size(600, 150, P2D)
    background(64)
    text_size(50)
```

In [3]:
```
def draw():
    # load numpy pixel array
    load_np_pixels()
    # create a gradient using numpy, assign to numpy pixel array
    np_pixels[25:125, :, 1:4] = np.linspace(0, 255, num=width)[:, None]
    # update numpy pixel array
    update_np_pixels()
    # oscillate the fill color between black and white
    fill((1 + sin(frame_count / 100)) * 127)
    # draw text to the screen
    text('this is py5!', 190, 90)
```

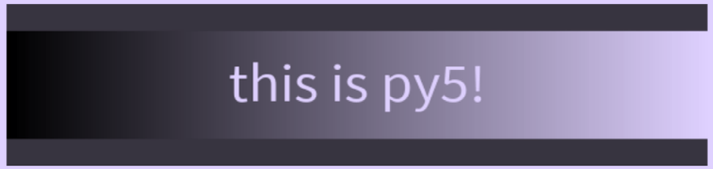
At its core, py5 is using the Processing Jar files to provide its functionality. It uses the Python library JPype as a bridge to connect CPython to the Java Virtual Machine.

In [4]:
```
run_sketch()
```

The Sketch appears in its own window. A screenshot can be embedded in this notebook.

In [5]:
```
py5_tools.screenshot()
```

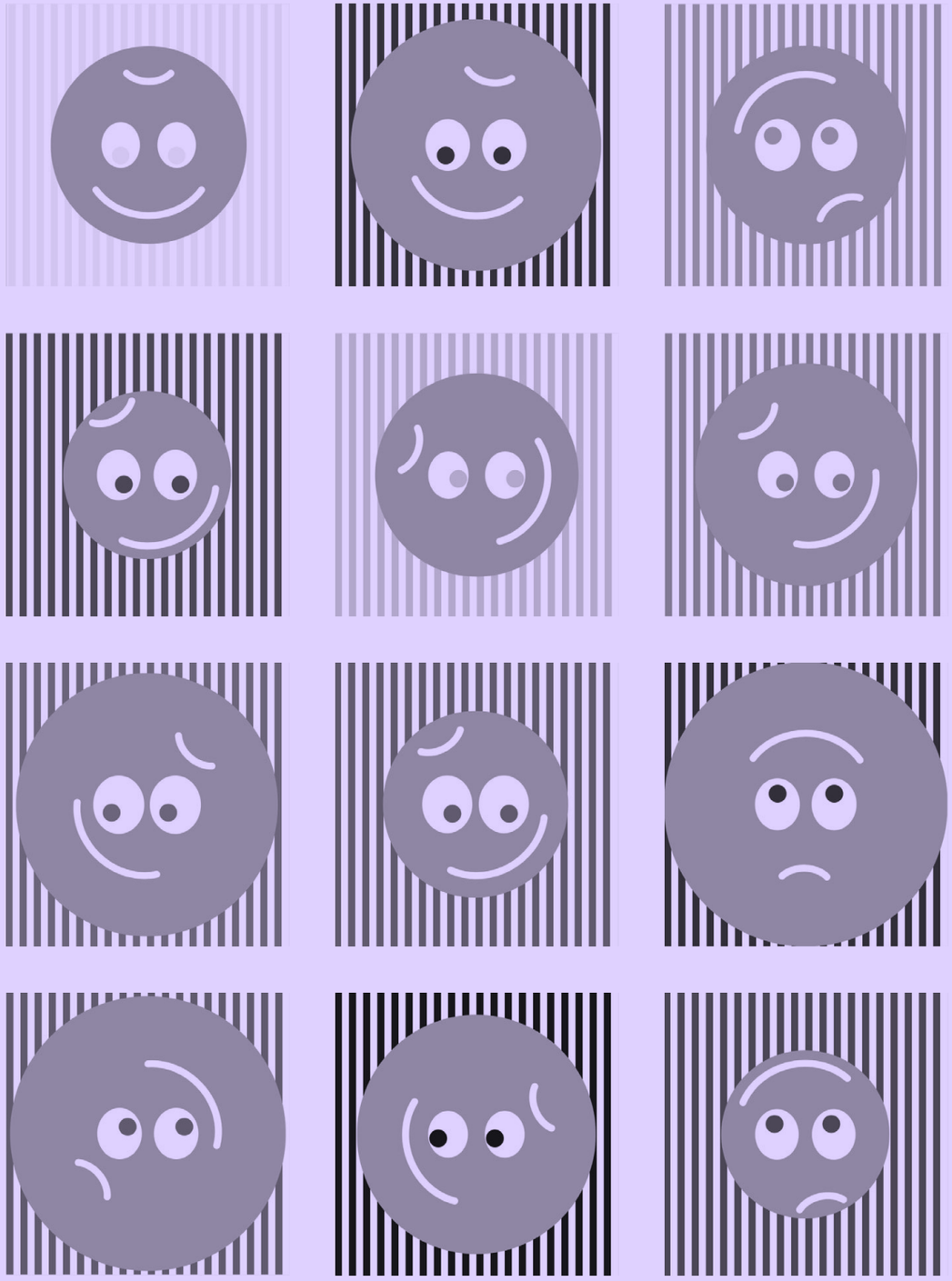
Out[5]:

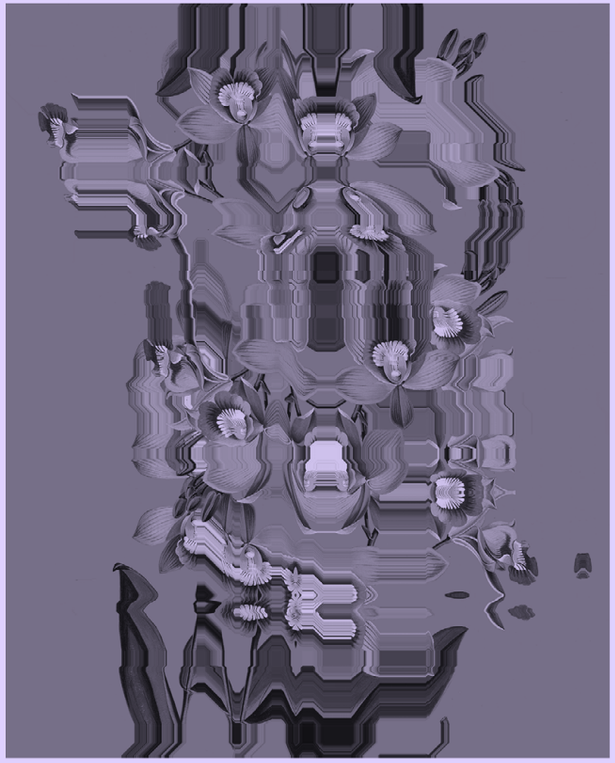


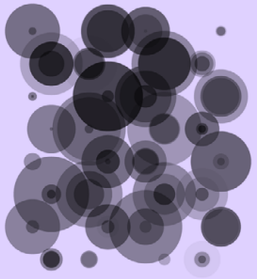

There is much more to learn about py5. Go to https://py5.ixora.io/ to get started!

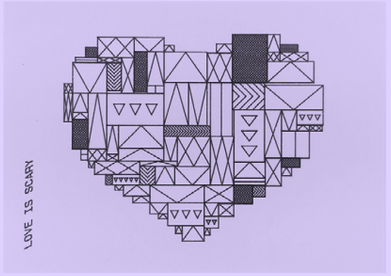
Questions? Comments? Reach out on Twitter at @py5coding or use the hashtag #py5

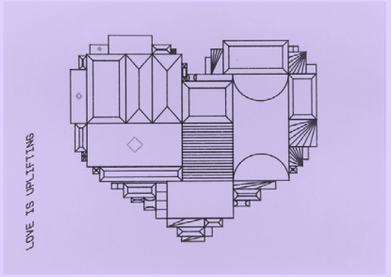


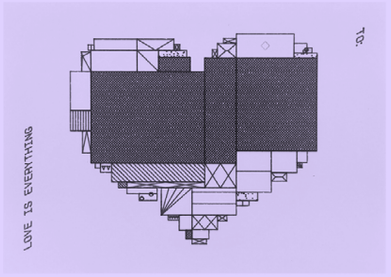


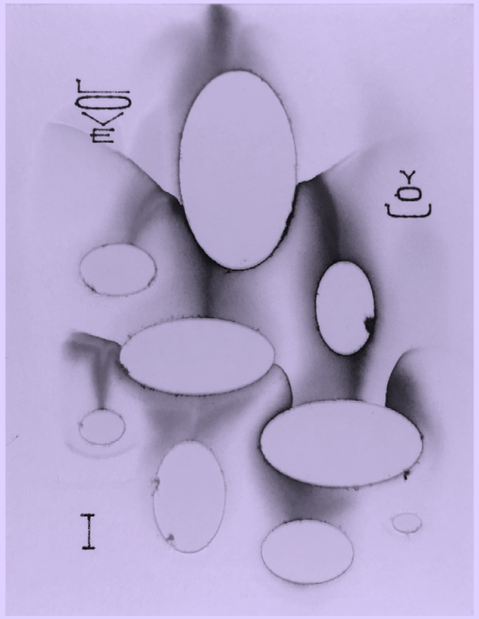




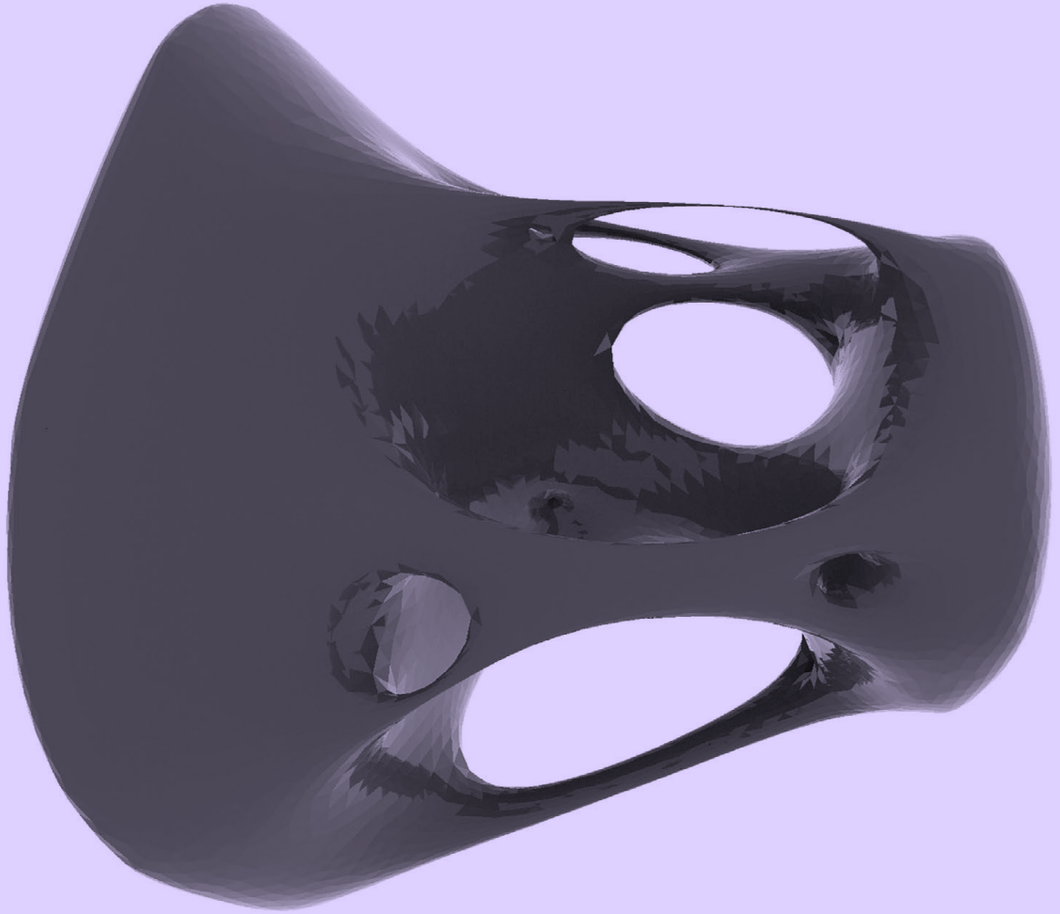

LOVE IS SCARY

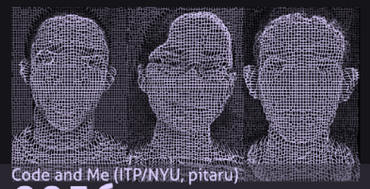

LOVE IS UPLIFTING


LOVE IS EVERYTHING


EXIT
EXIT





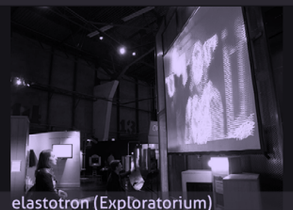


Code and Me (ITP/NYU, pitaru)
0056

refugees (Eyebeam/D-Fuse)
0076

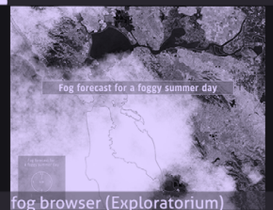
aperture (Eyebeam)
0092

elastotron (Great Wall of Oakland)
1.02

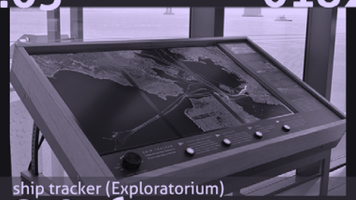


elastotron (Exploratorium)
1.05

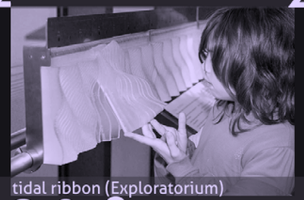
dividing space (Exploratorium)
0182




fog browser (Exploratorium)
2.0a4



ship tracker (Exploratorium)
2.0a6



tidal ribbon (Exploratorium)
2.0a8

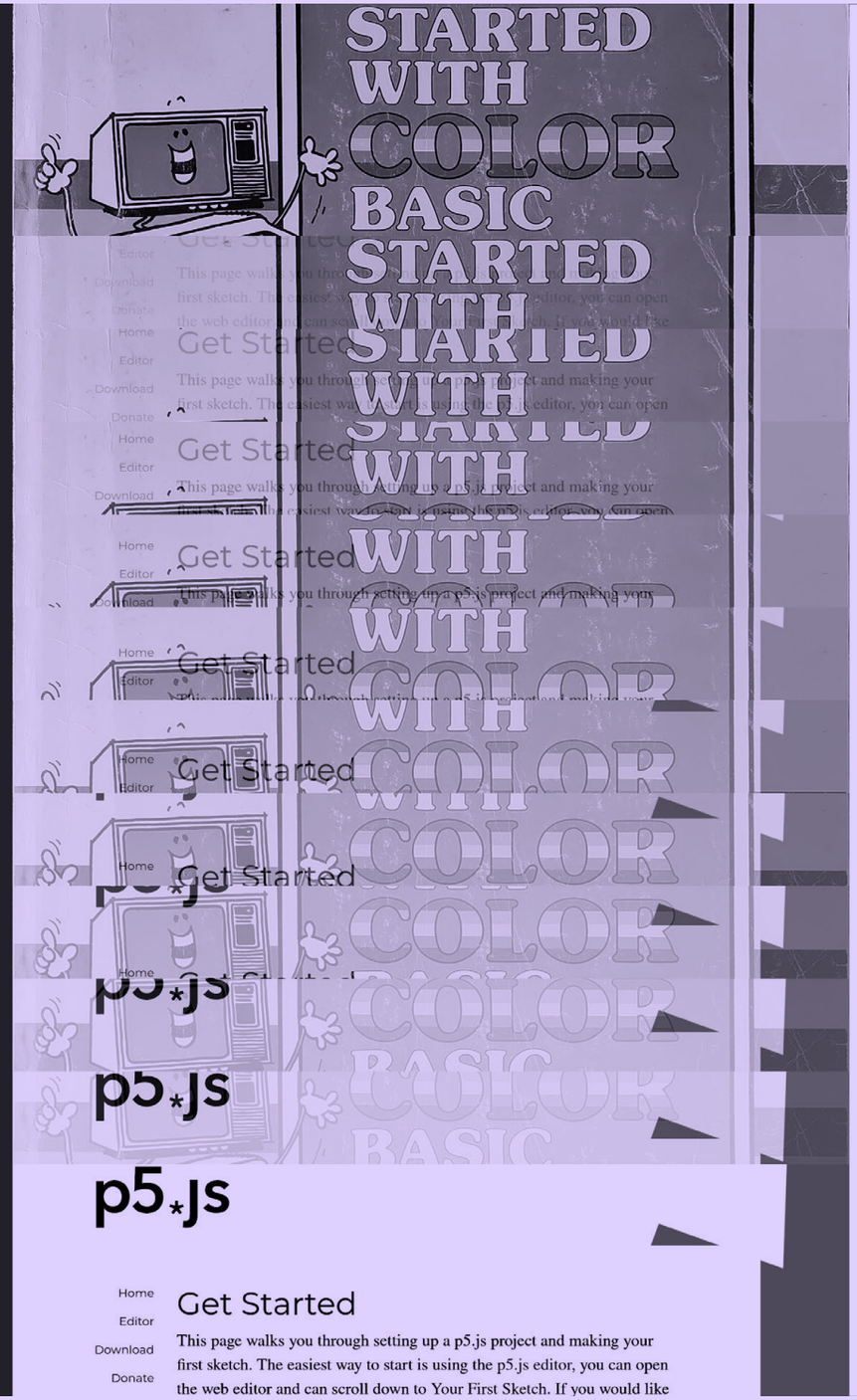
visualizing the bay (Exploratorium)
2.03

bubbles02a
1.2.0(p5.js)

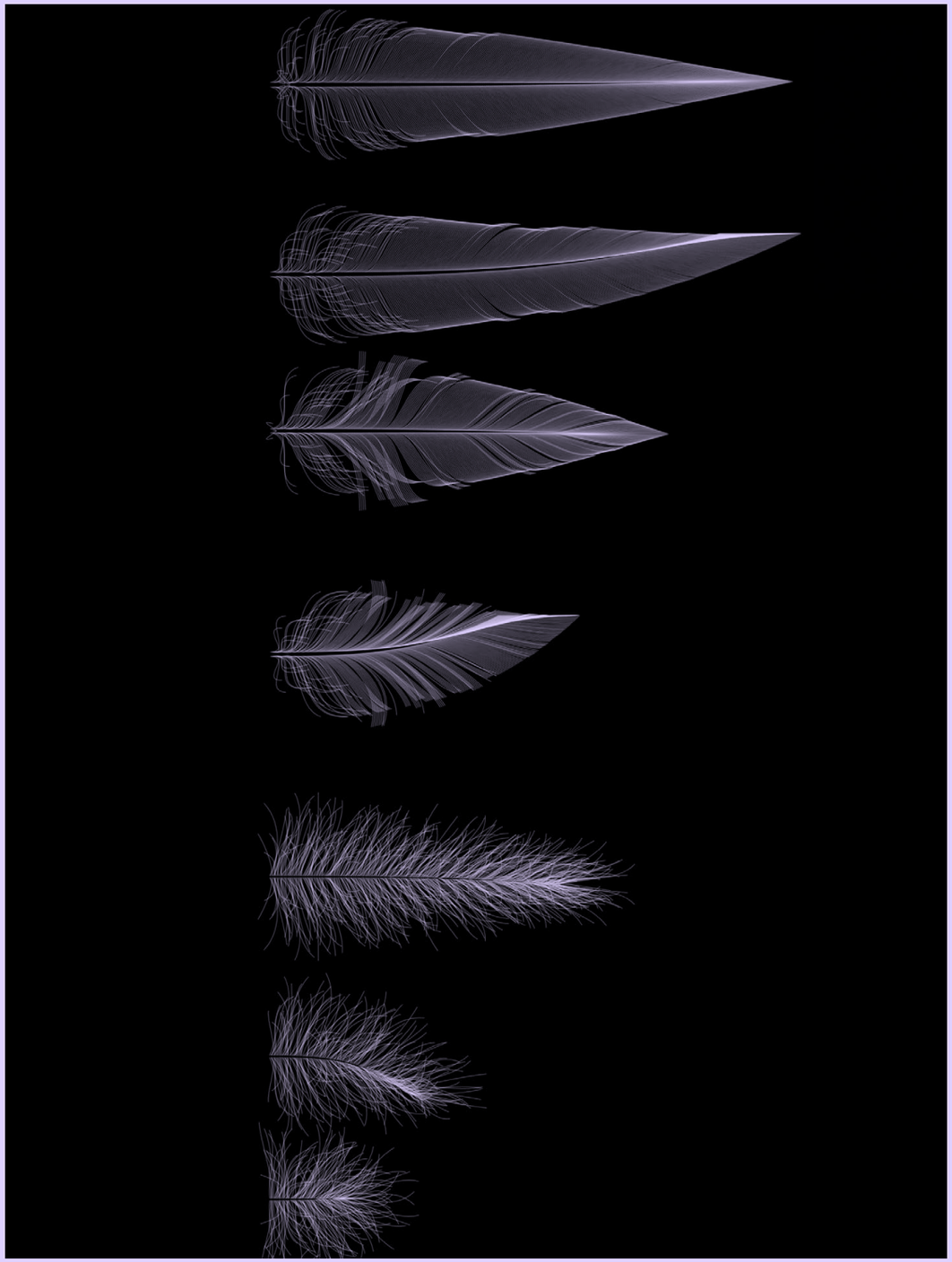
discgrid03
1.3.1(p5.js)



creativity progresses




by upgrading access






Ocean
Hello. My name is Senbaku.
- I live in Japan. I was a housewife raising my child, but on Christmas Day 2019, I met Processing, was made magic, and became a creative coder.
- I'm not aiming to write every day, but I enjoy writing code daily. I'm tweeting about it on Twitter. (https://twitter.com/senbaku)
- I used to draw illustrations using acrylic gouache and Illustrator as a hobby, and now I use code to draw those pictures. I enjoy expanding the range of things I can draw by incorporating math and science into my work.
- I wanted to contribute to the community, so I created a page to introduce books recommended by members of the creative coding community in Japan. (https://senbaku.github.io/) If you have any book recommendations, please let me know.

# My First Experience with Creative Coding.

Hello Legends! This is my experience with Creative Coding. If you are driven by coding and art then this is a perfect read for you.

I'm Sudhanshu Mukherjee and have done my engineering in computer science. I'm a Freelance Data Scientist and when I'm free I love reading about New Technologies, Creative art generation, Blockchain Tech, Cryptocurrencies and niche technologies which make their way in Tech.

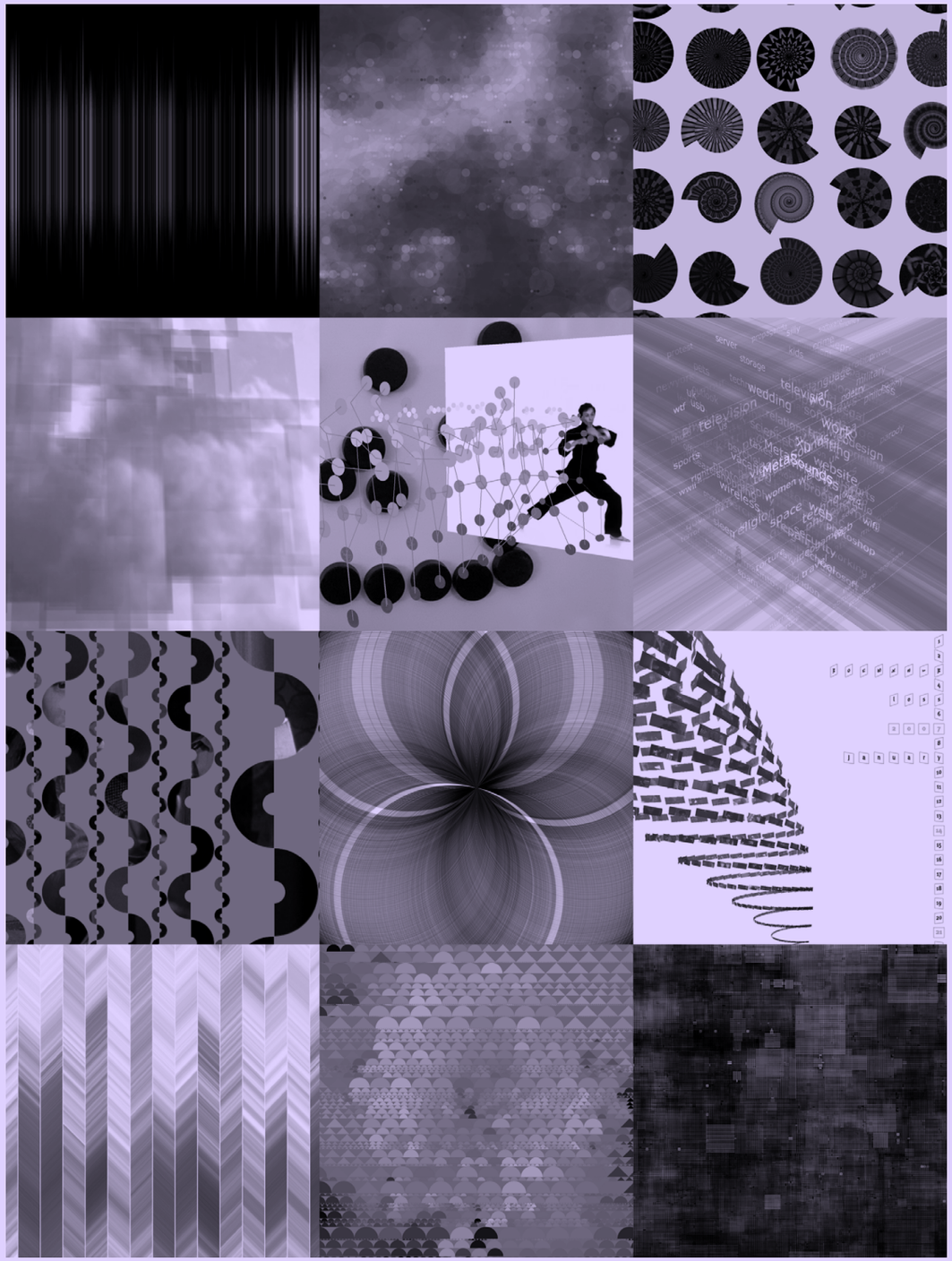
I never knew something called Creative Coding Art exists in this universe. I came across a workshop which was about generating music and visuals using code. To be honest I was scared coz I didn't have any idea about it. Keep everything aside I decided to go for it. The workshop was the most remarkable thing happened to me. It opened a different universe for me and it was so elegant that I spent next few days digging more about it. I was amazed that there exist a complete community who create and shares art with everyone. I really loved how everyone was so involved sharing each others' art and applauding every small thing which was created within community.

I think in my opinion we have so much more to grow this in this everyday growing technology. I want every developer to be familiar with the applications of p5.js. I'm grateful to Processing Community and p5.js for relentlessly working towards for educating more and more  enthusiasts to ignite interests of everyone.

I also feel there's so much more to explore for me in this and it would take millions of baby steps to reach there. My heart is filled with so much of confidence writing this. I'm very sure there will be a day where creative coding is pursued as a hobby and every kid would spend hours doing it.

Sudhanshu Mukherjee
Data Scientist
Instagram - @senseiwhocodes
Github - sudhanshumukherjeexx
Linkedin - www.linkedin.com/sudhanshu-mukherjee

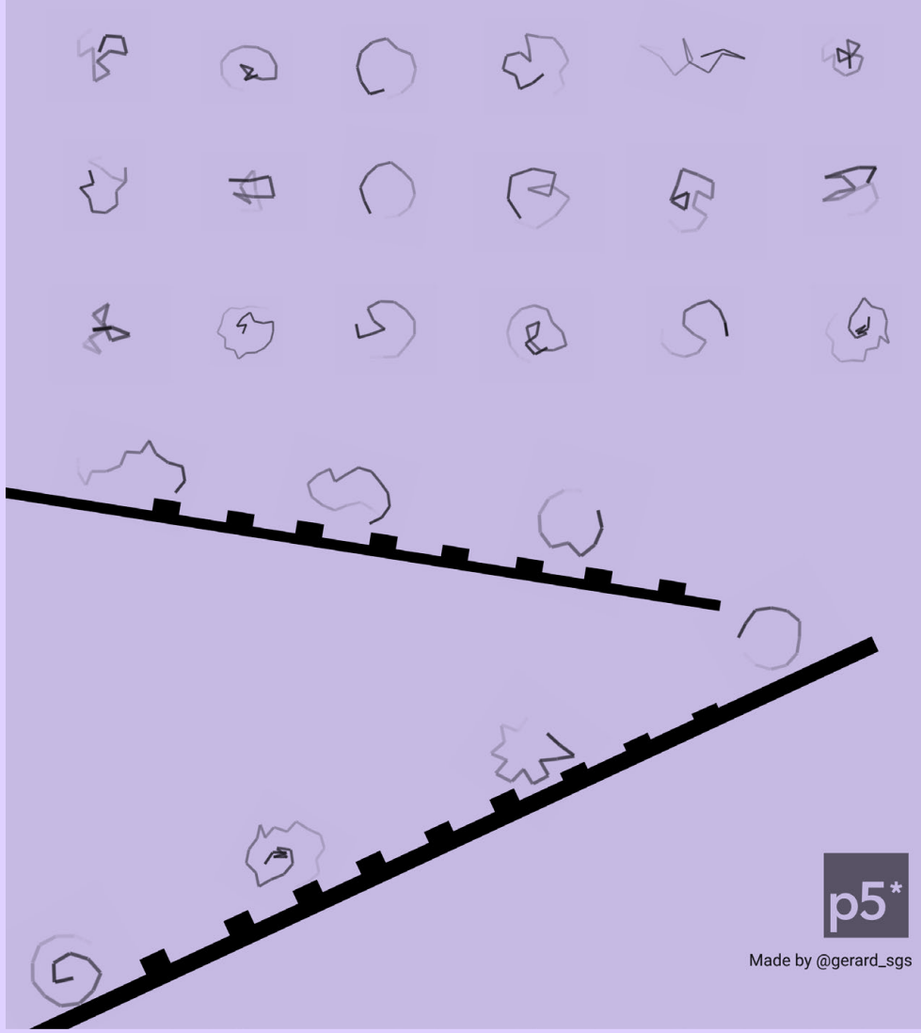




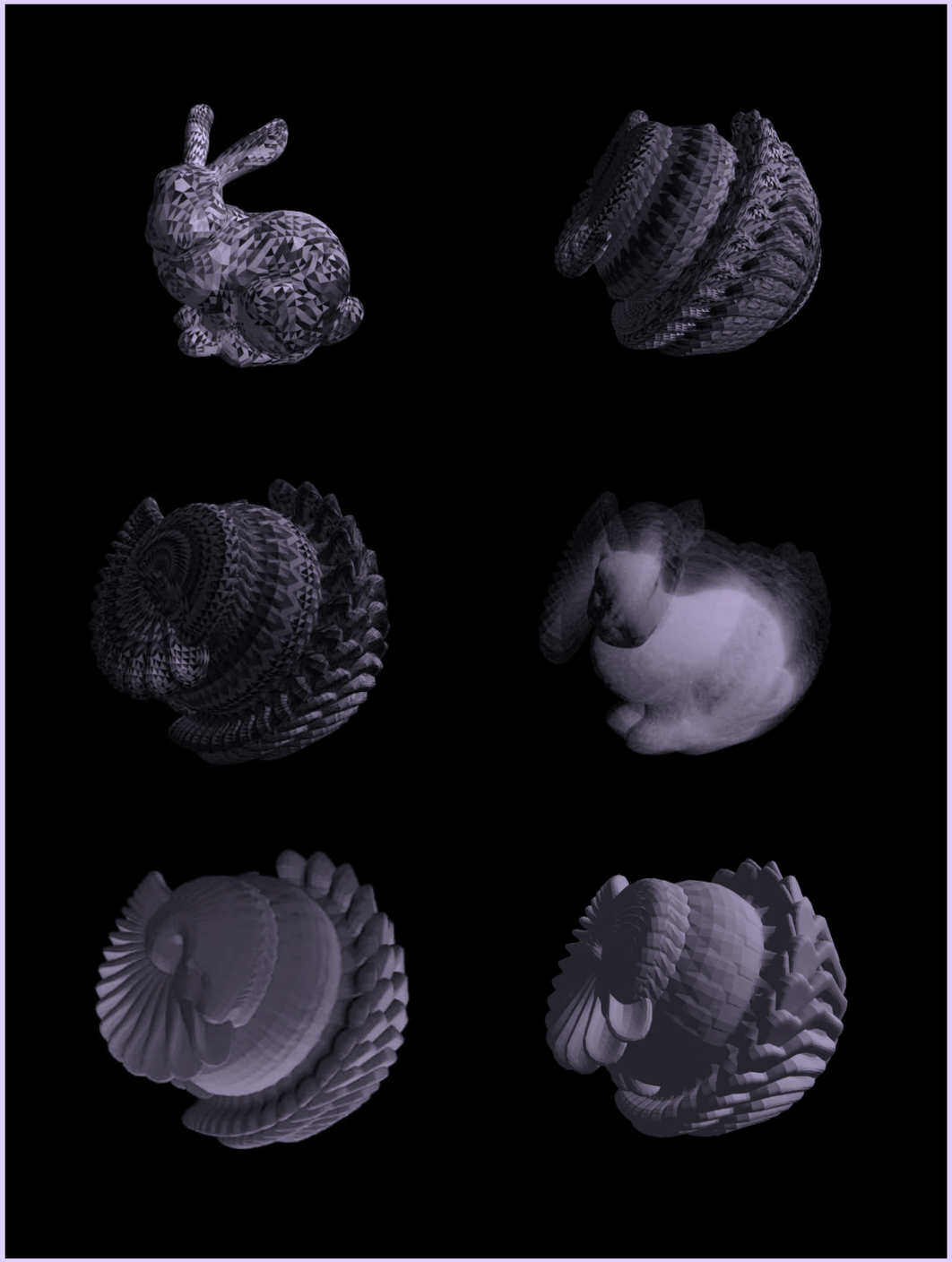
# COEVO

A 2D physically-based environment with artificial agents that create 2D solutions for given problems

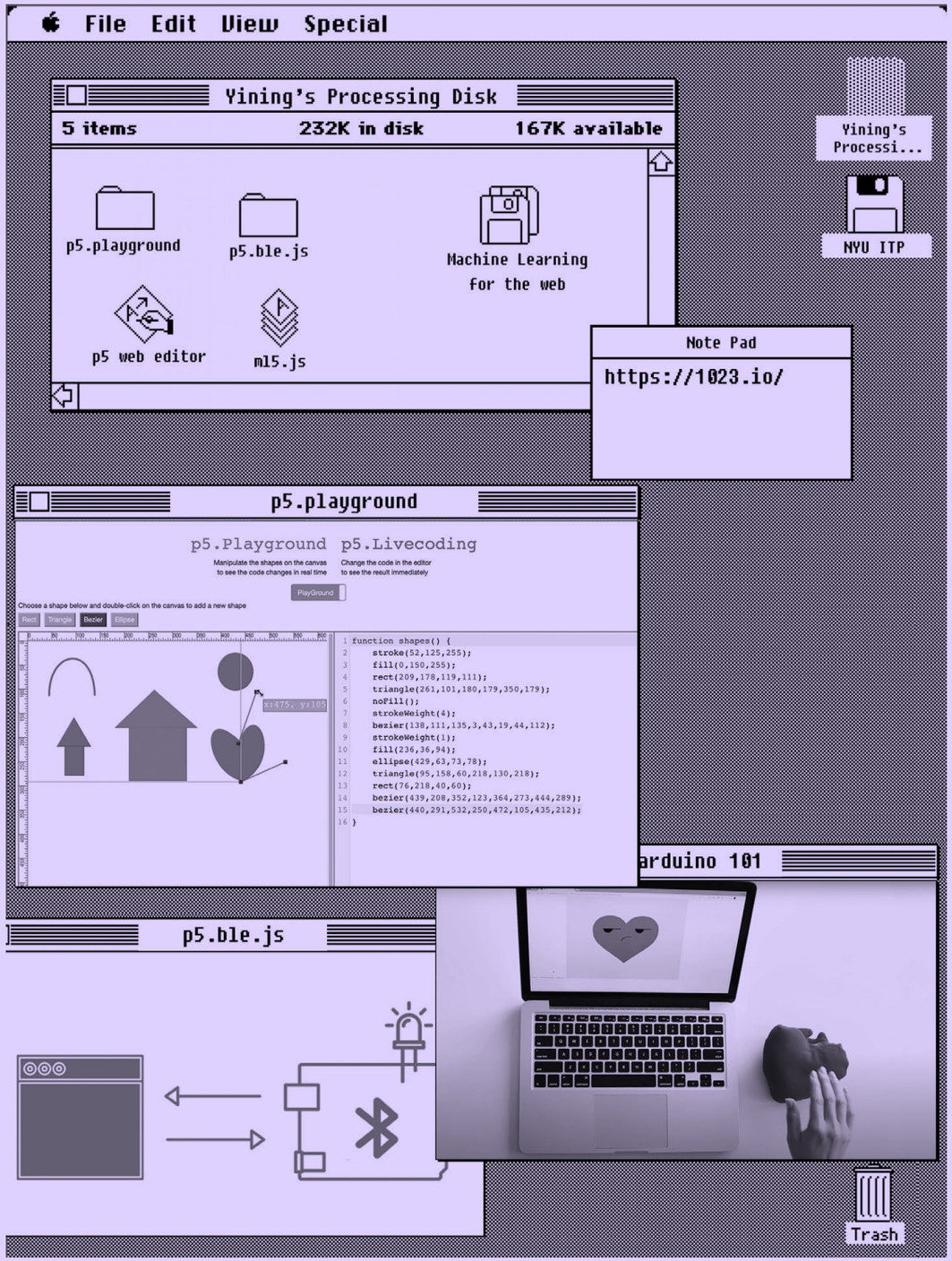
“Create an object that moves through an inclined plane”








File Edit View Special
Yining's Processing Disk
5 items
232K in disk
167K available
p5.playground
p5.ble.js
Machine Learning for the web
p5 web editor
ml5.js
Yining's Processi...
NYU ITP
Note Pad
https://1023.io/
p5.playground
p5.Playground p5.Livecoding
Manipulate the shapes on the canvas to see the code changes in real time
Change the code in the editor to see the result immediately
PlayGround
Choose a shape below and double-click on the canvas to add a new shape
Rect Triangle Bezier Ellipse
function shapes() {
stroke(52,125,255);
fill(0,150,255);
rect(209,178,119,111);
triangle(261,101,180,179,350,179);
noFill();
strokeWeight(4);
bezier(138,111,135,3,43,19,44,112);
strokeWeight(1);
fill(236,36,94);
ellipse(429,63,73,78);
triangle(95,158,60,218,130,218);
rect(76,218,40,60);
bezier(439,208,352,123,364,273,444,289);
bezier(440,291,532,250,472,105,435,212);
}
arduino 101
p5.ble.js
Trash

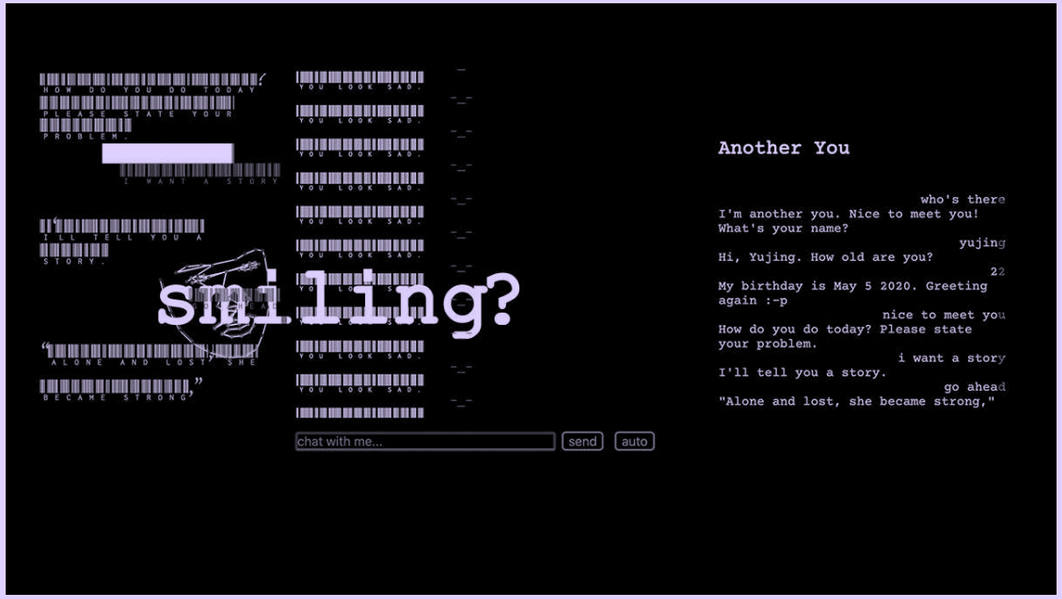
# Another You

## Artificial Companion

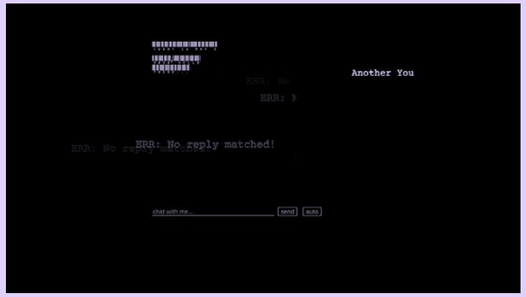
*Another You* is an artificial companion which represents another you. 'It' is the negative side of you, another version of you. It is depressed, sad, and passive. Based on the expressions on your face, it will automatically generate an avatar looks like yourself and have conversations with you.

It chats(in text & sound) with you and provides negative thoughts to you. This another you also has the power to decide when to end the conversation and when to start. It can recognize some of your facial expressions or gestures and give some suggestions.

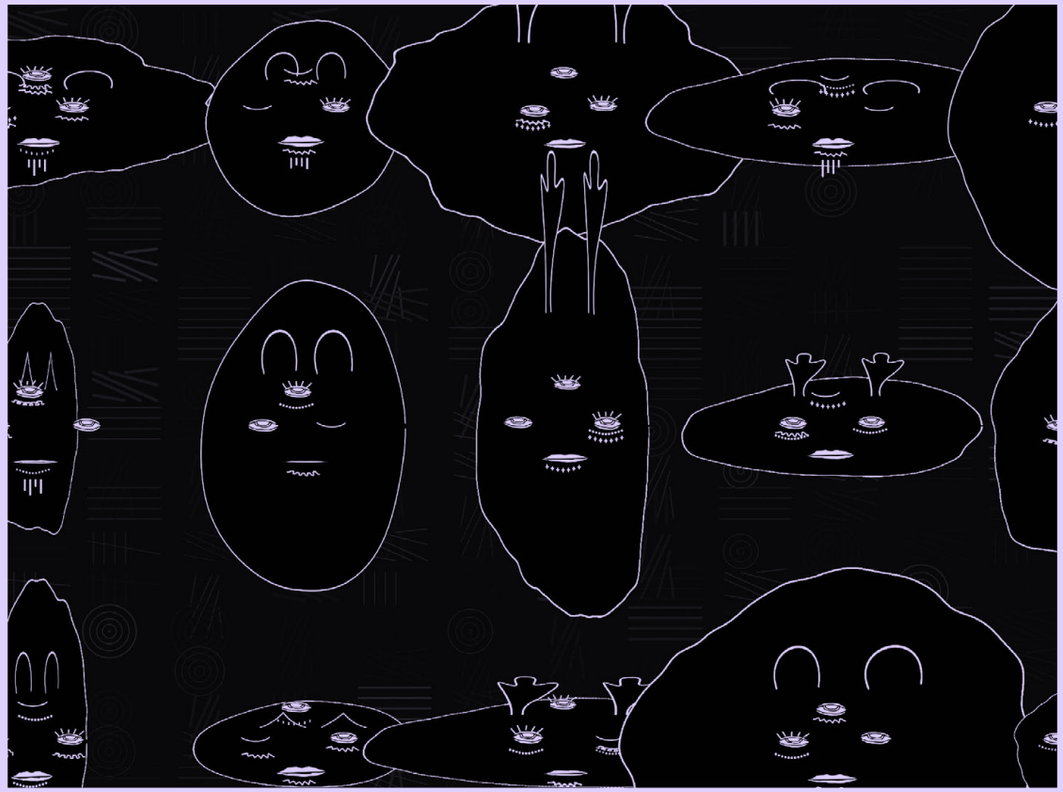
It is not a controllable 'pet' that you can in charge of. And it is also not a happy companion.









https://cc-portfolio-munus.glitch.me/p2.html

Unisex Face Generator, Munus Shih (2021)

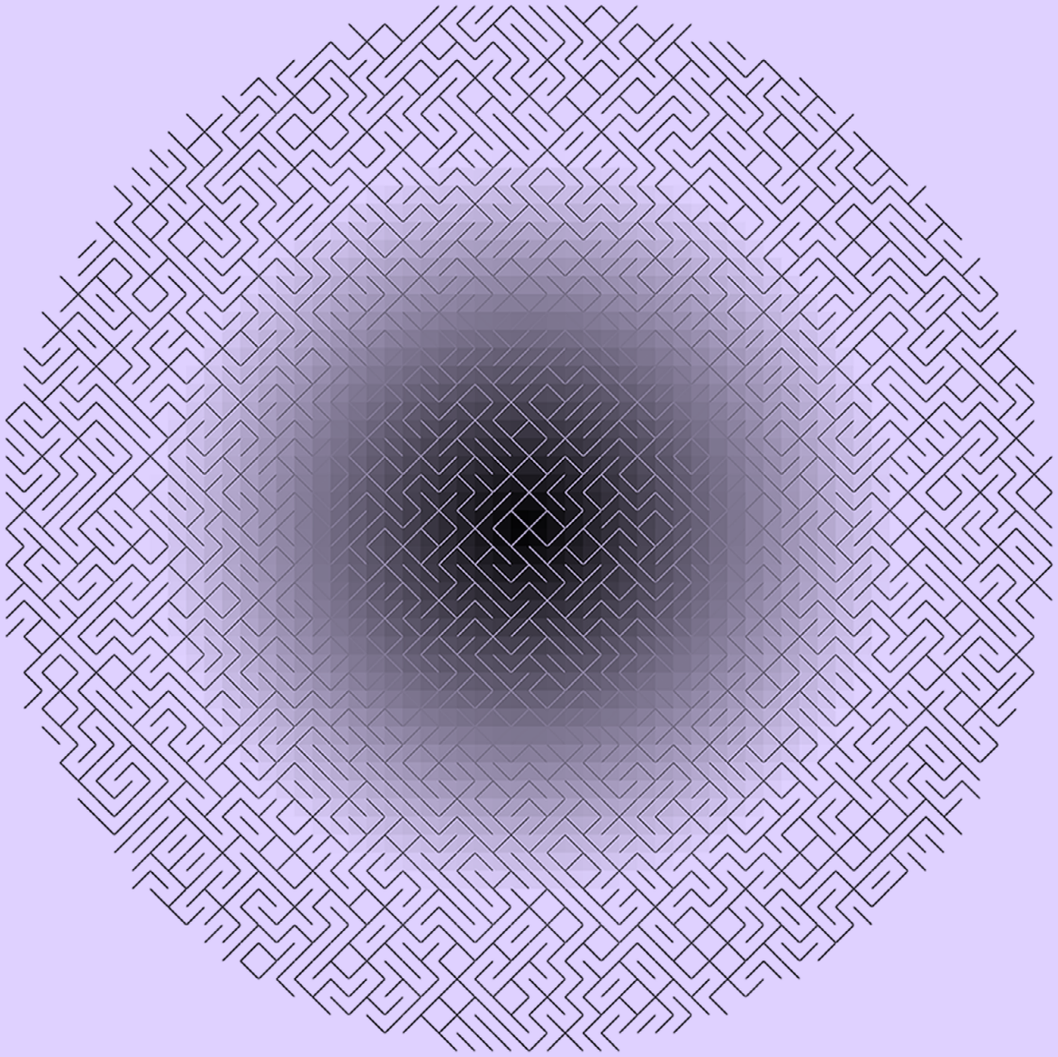
I personally love to draw faces, and this is my first attempt to transform my drawing into a random generator. This piece of work is B&W because I wanted the viewers to notice that squiggly, hand-drawn style in my lines using random functions.

Not only generating the faces using the random function but also their names are generated randomly from a list of thousand unisex names on the internet. Every one of them generated from the random function has a unique name and is part of the new school, thus the font I used here is our customized new school font. They would introduce themselves every time you create them.

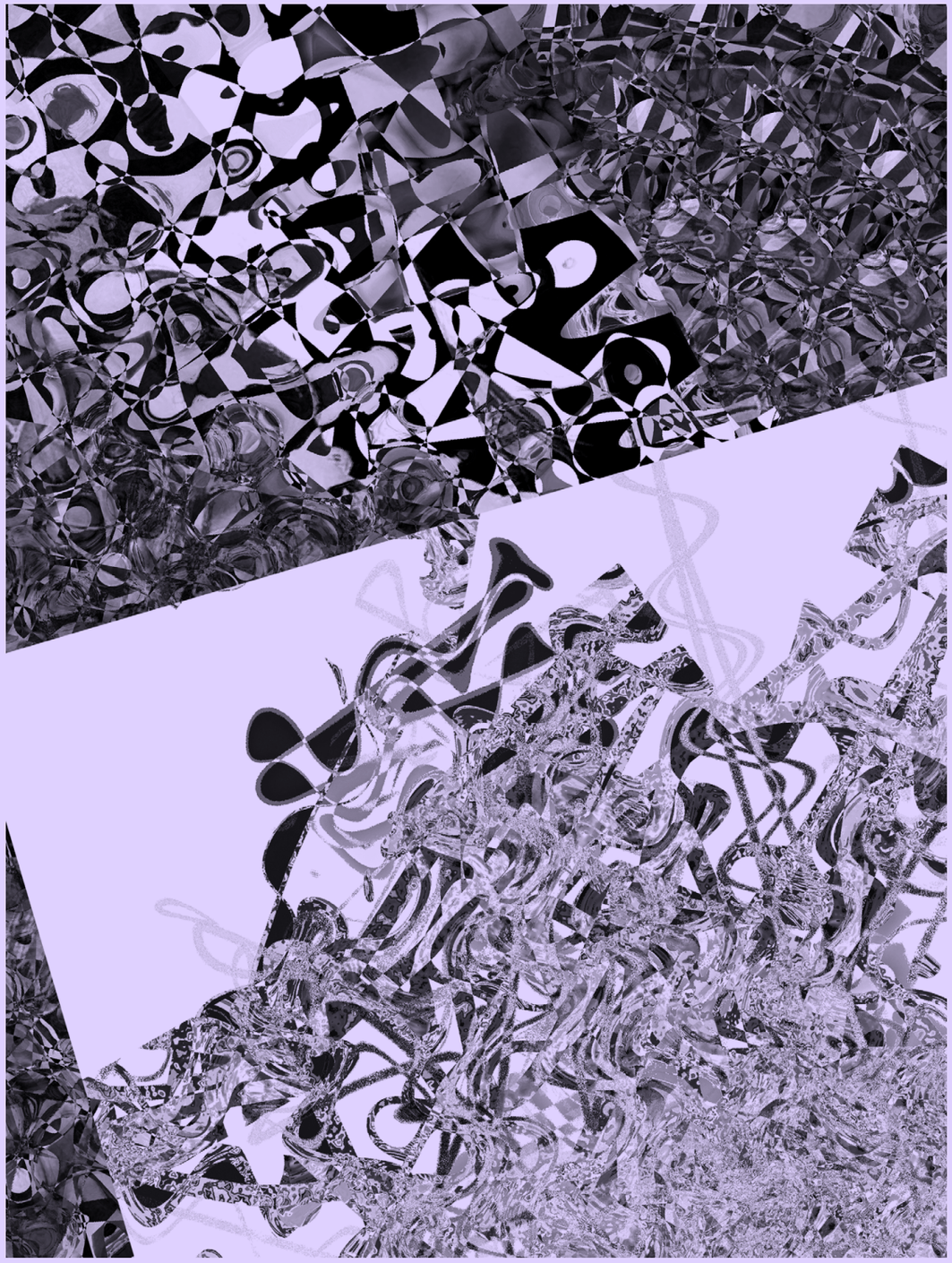
I love to sketch, but I didn't spend much time on drafting this time. I was trying to implement the idea of "coding as painting", and just get my hands on the coding process without thinking too much. Change the concept as I'm coding new things, get into 'the zone'.

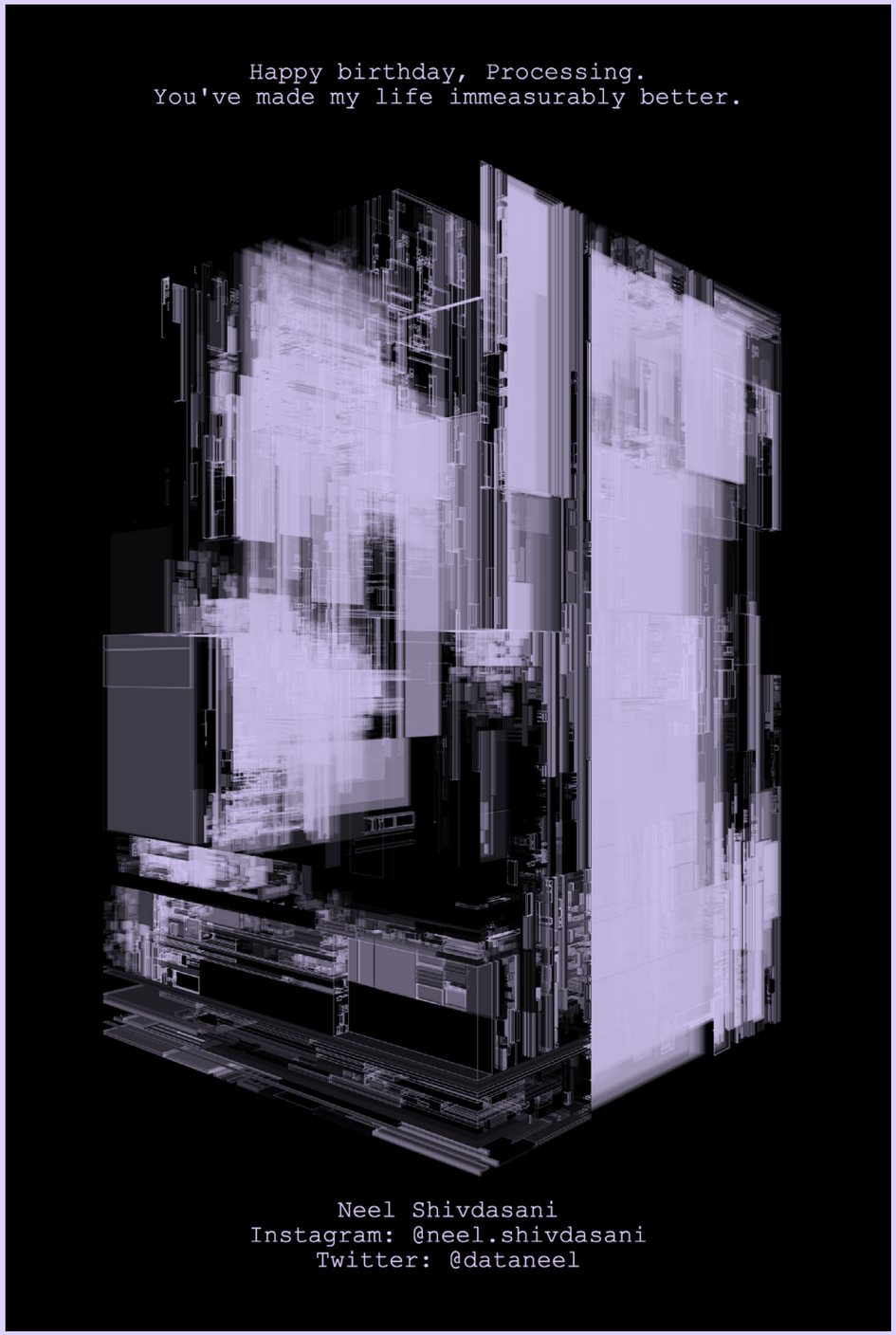
I first started with the face, and gradually add more features to it. Before finishing the whole thing, I feel like there's something missing. What else do we usually think of when we think of someone's face? What is something that makes this face unique among all the faces we've seen. Then, ding dong! The idea of generating their names randomly popped into my head, and I soon found this library called 'p5.speech' to take my idea to the next level.

Not only did I successfully pulled their names from a list of a thousand unisex names, but I also use p5.speech to randomly assign them their speech pitch, speech rate, and greeting words.

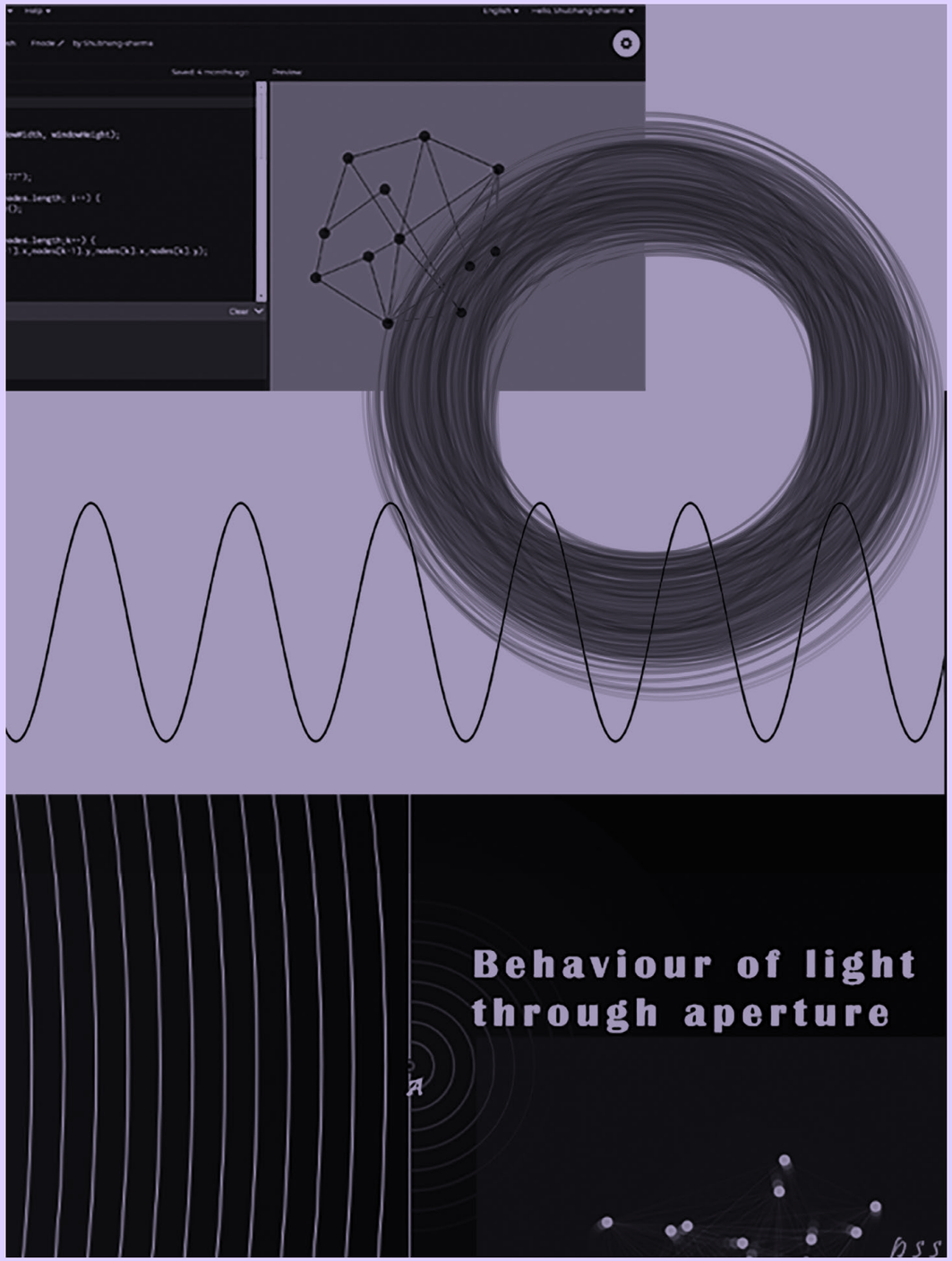






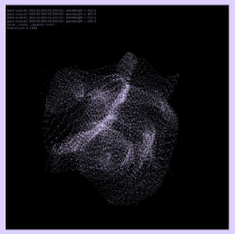
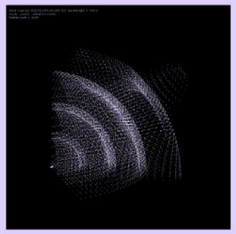
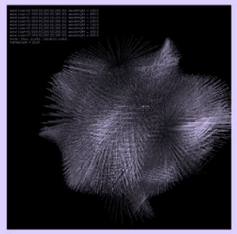

Happy birthday, Processing.
You've made my life immeasurably better.
Neel Shivdasani
Instagram: @neel.shivdasani
Twitter: @dataneel




Behaviour of light
through aperture

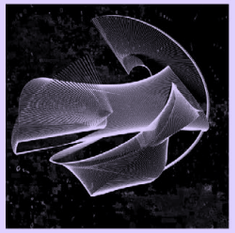
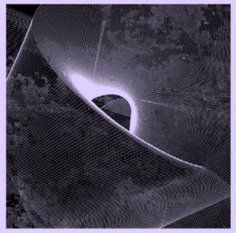
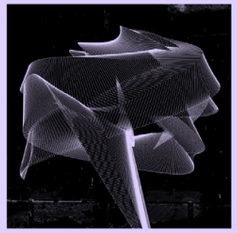
# HISATOMI SHUJI

私は日本の高校生で、2020年にProcessing(java)を使い始めました。私は物理が好きで、物理法則を可視化するのによくProcessingを使っています。
I am a high school student in Japan. I began to use processing in 2020. I like physics and I often use Processing(java) to visualize some Physics Formulas.

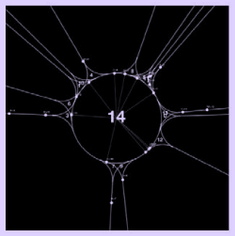
"Wave Interference Simulation"
-2021.02.17

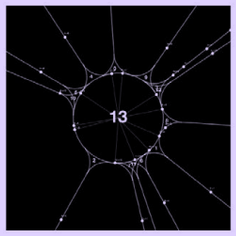
波源の個数、波長、周波数、座標を変更できる。3次元空間における波の変位を計算し、点や直線で表現する。
The number of wave sources, wavelengths, frequencies, and coordinates can be changed; the displacement of waves in three-dimensional space can be calculated and represented by points and lines.

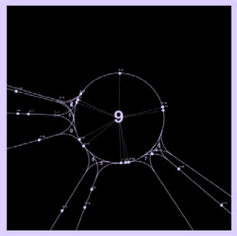




"Vegetative State"
-2020.11.03

この作品は私の高校の学園祭「音楽と展覧の会」のために制作されたもの。2020年の社会状況を表現した映像作品である。
This work was exhibited at the school festival "Ongaku to Tenran no kai (Music and Exhibition)" of my high school. It is a video work that expresses the social situation in the year 2020.

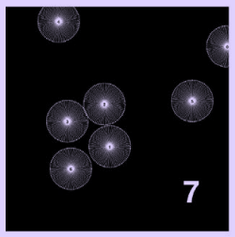


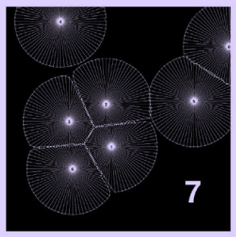


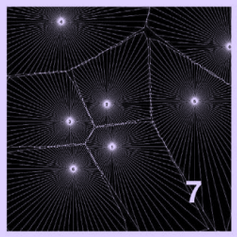



"Roundabout"
-2021.06.13

文字通り「ラウンドアバウト」。
Literally "Roundabout".



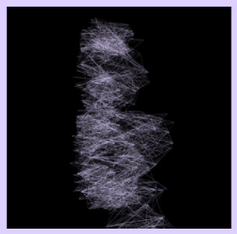





"Districts"
-2021.06.01

ランダムに発生する平面上の点から、領域が拡大していく映像作品。
This is a video work in which several areas expand from randomly generated dots on a flat surface.



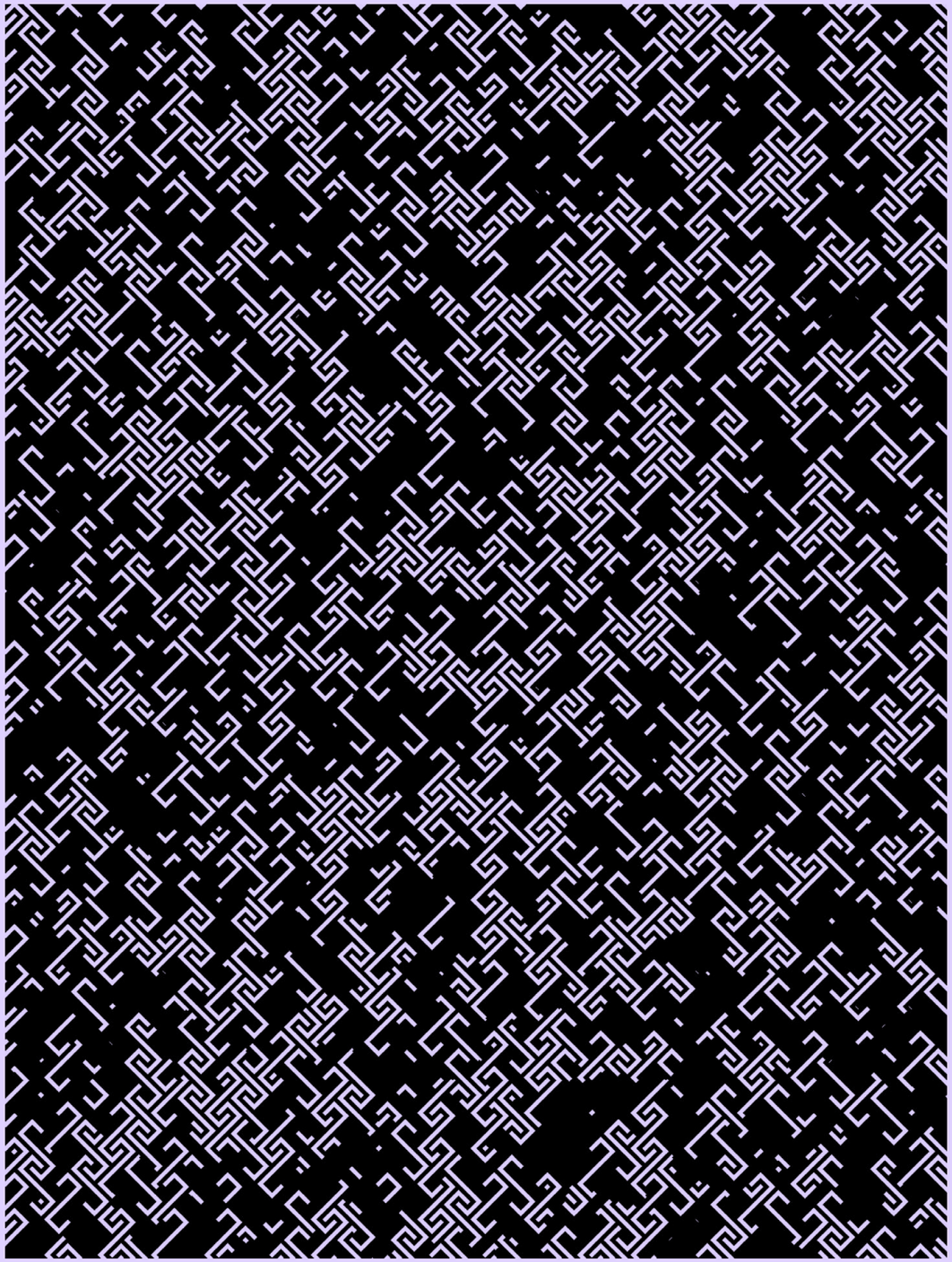


"Lines & Alphabets"
2020.12.18

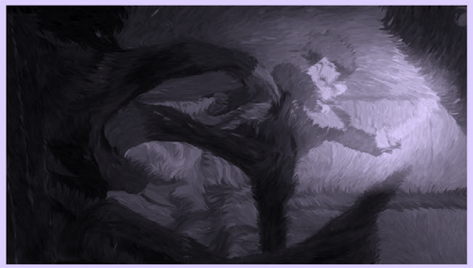
アルファベットを点と直線で表現する映像作品。
A video work in which the alphabet is composed of dots and straight lines.

If you like my works, please let me know on Instagram (@shuji15.9041)





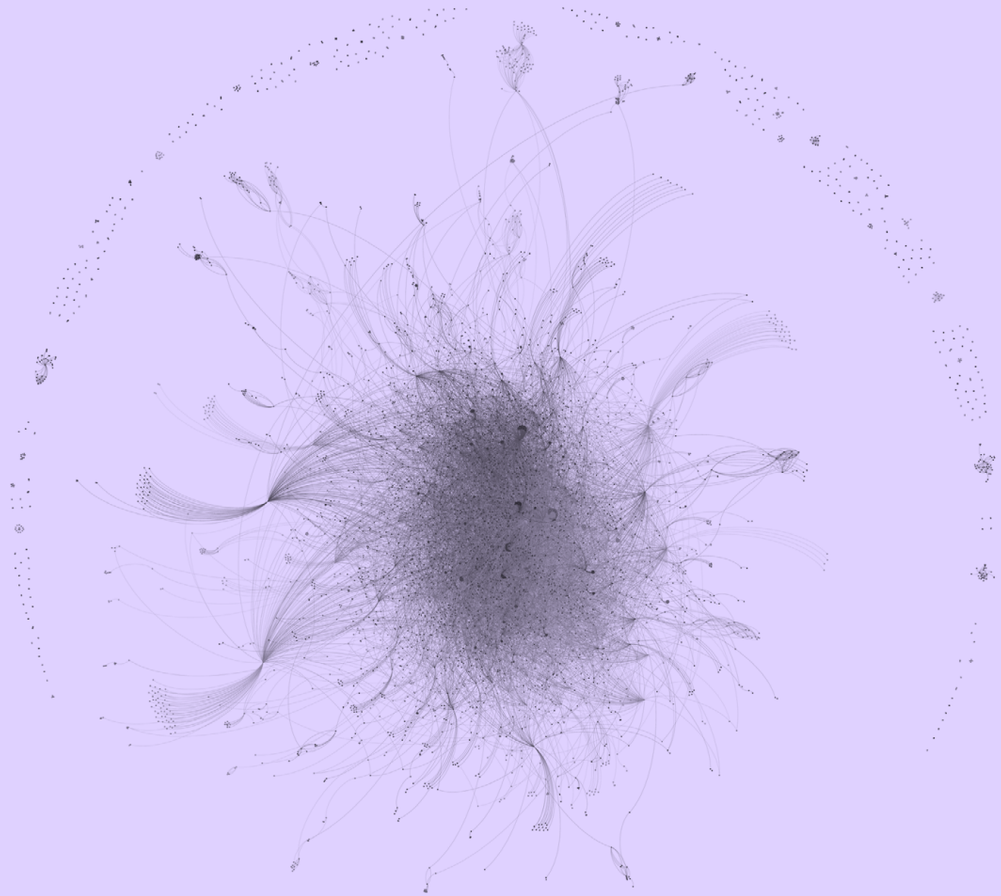





Carrie Sijia Wang, *The System 2.0*, Live-Streamed Performance, 2020

I created this performance using p5.js during the 2020 lockdown in New York. Being part of the Processing Community has enabled me to rethink my relationship with technology, the internet, and all the other systems embedded in my everyday life.





# Re: a community that moves us forward

As “social media” was hitting the headlines to confuse us with its purpose, the Processing community inspired me to imagine something better.

I launched OpenProcessing in April 2008. Having shared over a million open source projects since, an intertwine of artists, coders, teachers, and students proved what an exceptional community we are.

One that creates, inspires, and moves forward. Together.

Cheers to the next 20,

Sinan Ascioglu

Visualization of the community interactions between 2008-2016 on OpenProcessing.org. Created by John Davies using Gephi at Nesta Foundation, UK, 2016.
https://www.nesta.org.uk/blog/gallery-how-digital-tech-changing-art-and-design/

# FROM AN ARCHIVE OF WEBGL / 3D CONTRIBUTIONS IN P5.JS

**rough pass at webgl renderer and basic line drawing** Browse files

main (#598)
v1.4.0 ... 0

**indefinit** committed on Dec 16, 2014
1 parent 8a7eb4a commit 96bf886143d073e1124a061092c91341384b997c

**✓ updates 3d sketch example with fps stats, first attempt at reviving 3...** Browse files

…d primitives cube

main (#644)
v1.4.0 ... 0

**indefinit** committed on May 27, 2015
1 parent 6191ab0 commit 3aa5751c4fa07f02fc008fc1fb9890260f035597

**merges in sphere and cube primitives from karenpenglabs** Browse files

main
v1.4.0 ... 0

**indefinit** committed on May 31, 2015
1 parent 27fa3a0 commit f56d09b776010aac29b78693f3aa75f429eeba1d

**implements video as texture in webgl: WIP still buggy** Browse files

main (#889)
v1.4.0 ... 0

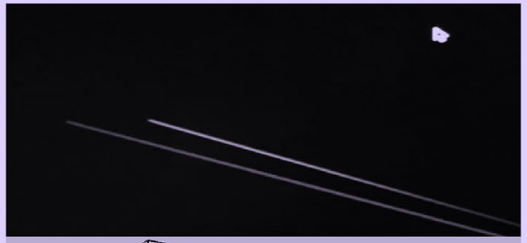
**indefinit** committed on Aug 25, 2015
1 parent 30a9d73 commit f89a81b0f732df92ed0e6f80a753a4846ab64414

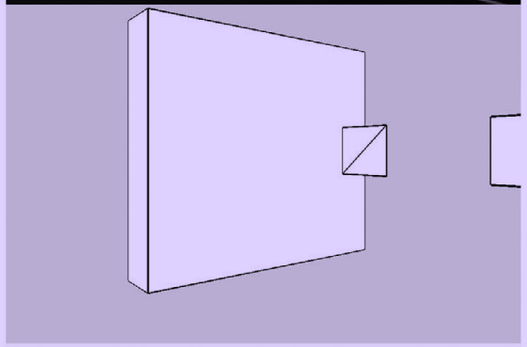
**✓ Merge pull request #1285 from mindofmatthew/3d-model-loading** Browse files

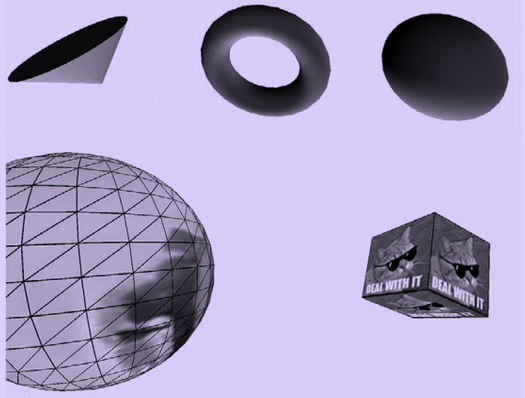
3d model loading

main (#1370)
v1.4.0 ... 0

**indefinit** committed on Mar 5, 2016
2 parents 7da787b + 2802b42 commit c020ccbee069971bffb52e548cdf7e0f34c

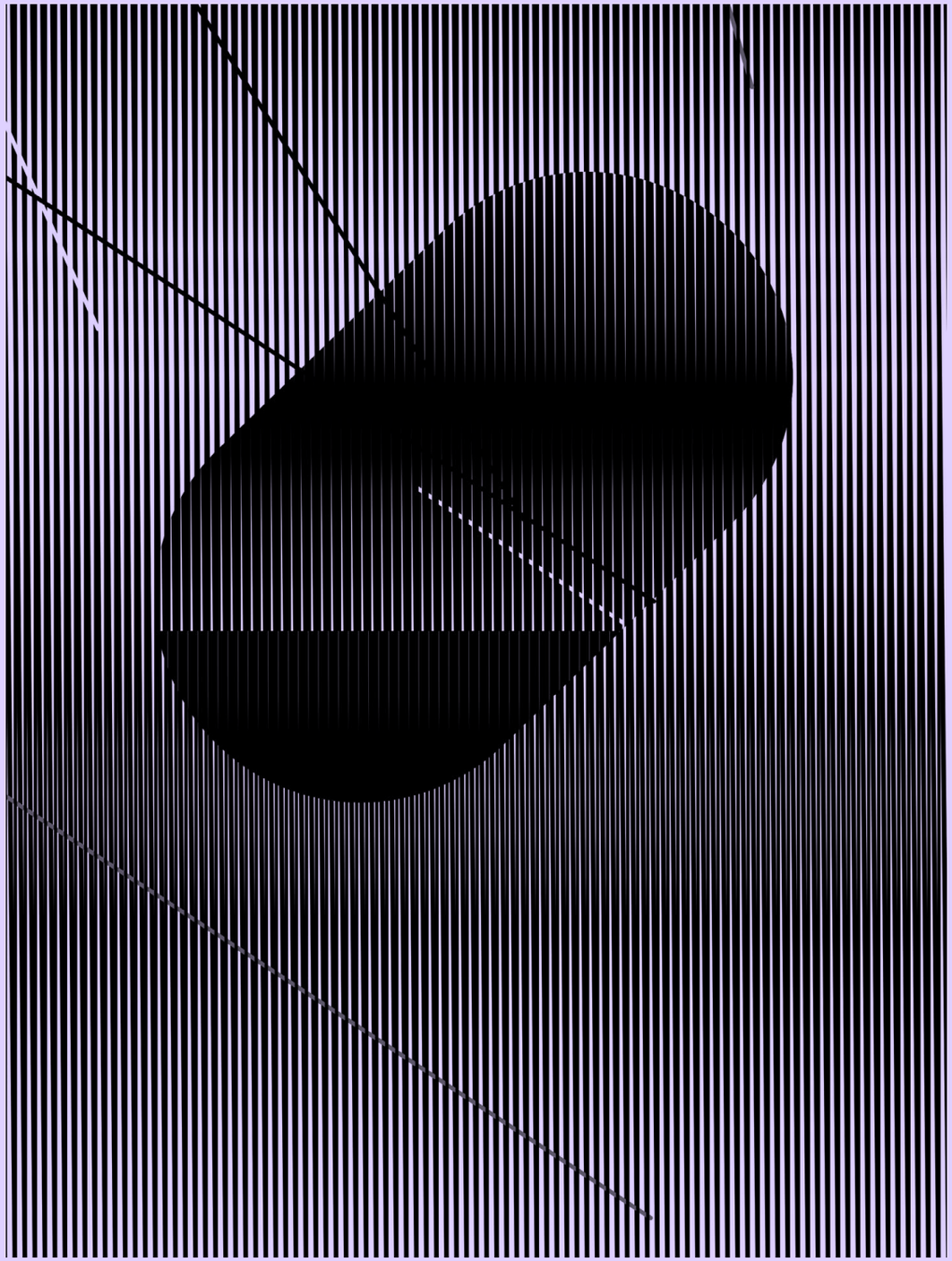


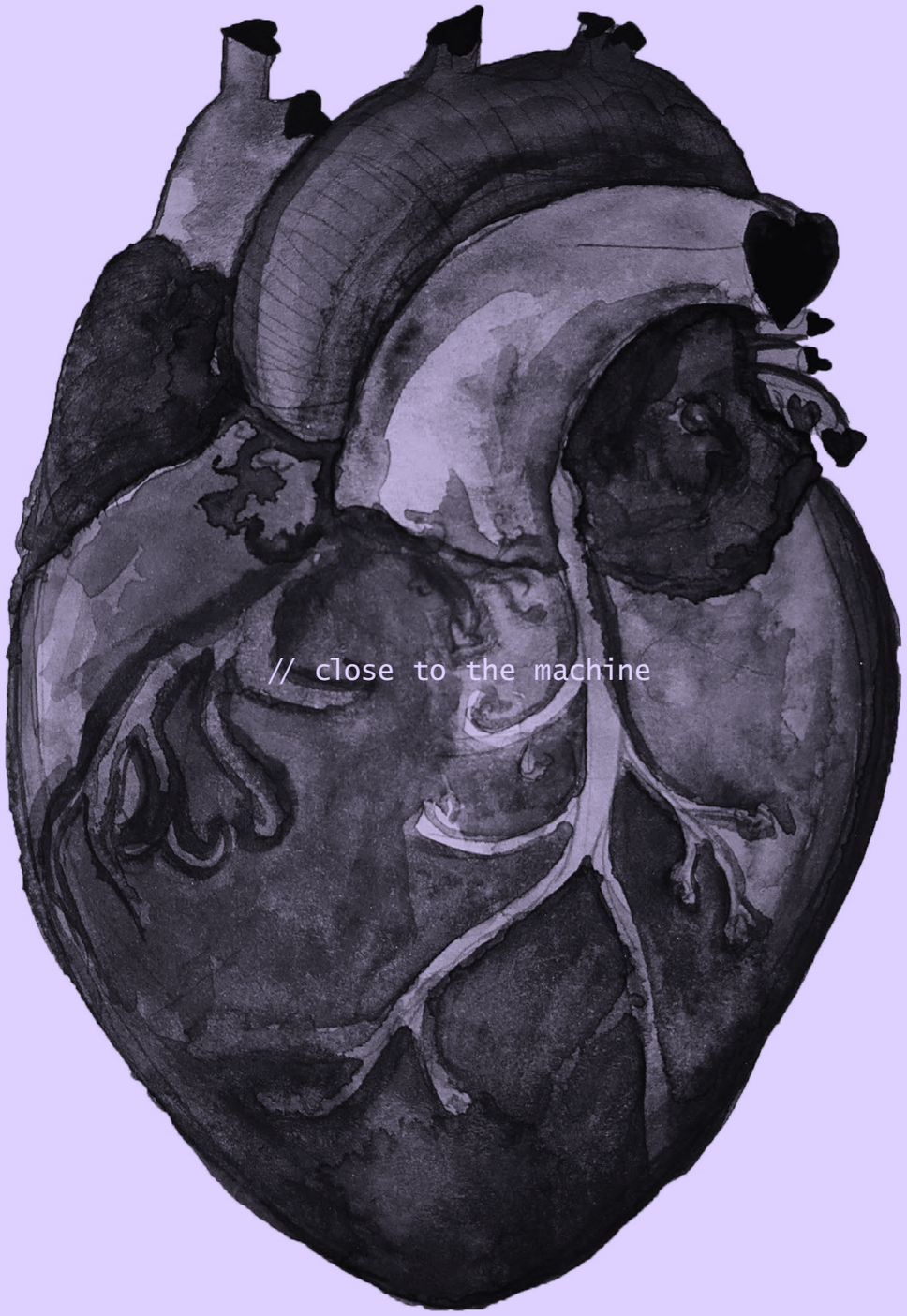




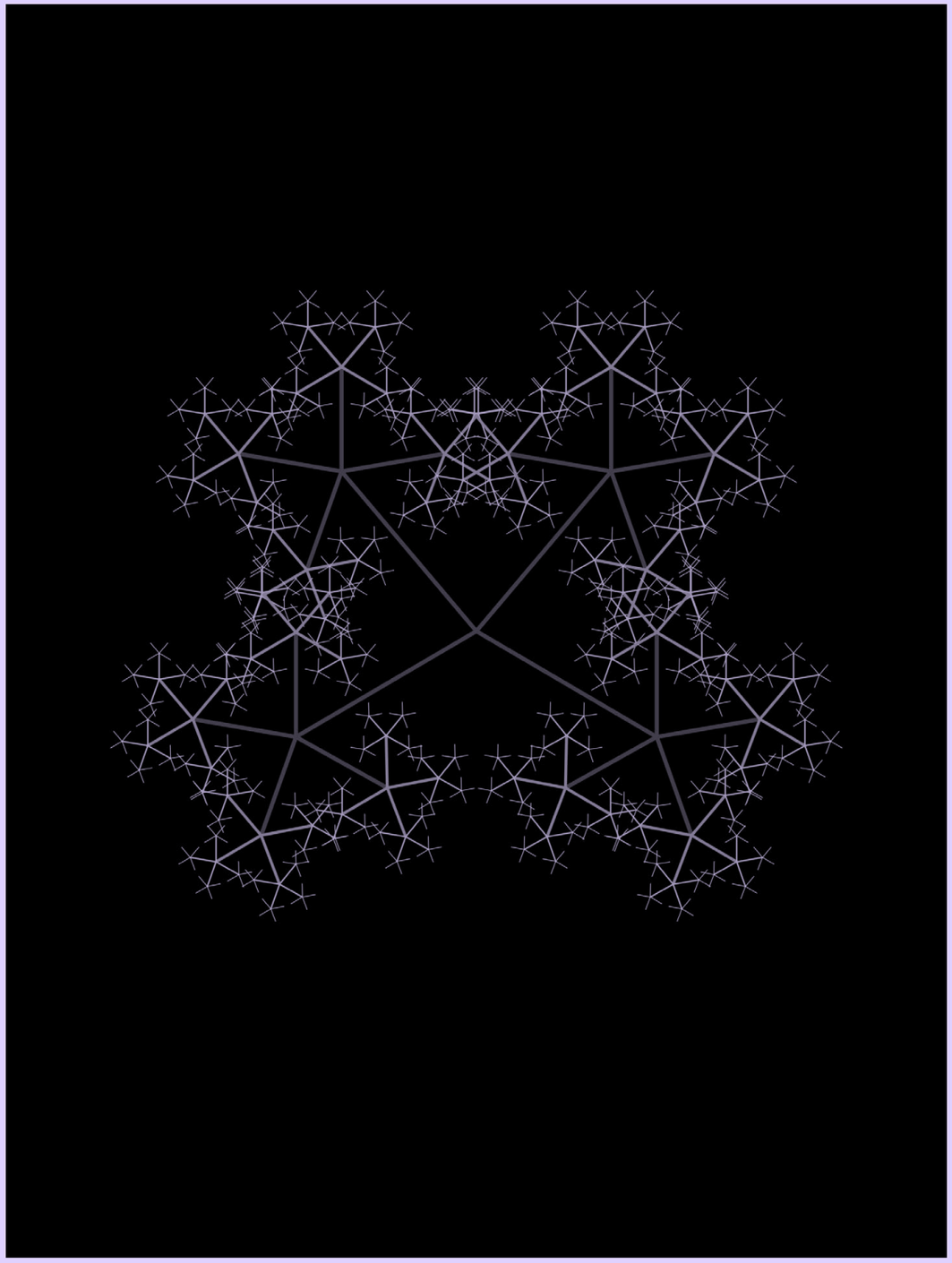

I started laying the ground work on the p5 3D renderer (webgl) in late 2014, and from 2015 to 2016 was joined by Karen Peng and Matthew Kaney, as we collaborated on lighting, texturing, and 3d model loading features. It was exciting to be involved in p5js, and more broadly the Processing Community, because we were all learning and growing together. It felt a liittle like building a house we could all live in, where someone would start constructing a floor or a room, and we'd keep adding onto and around it—desperately trying not to break anything in an unforseen location. -Kevin Siwoff

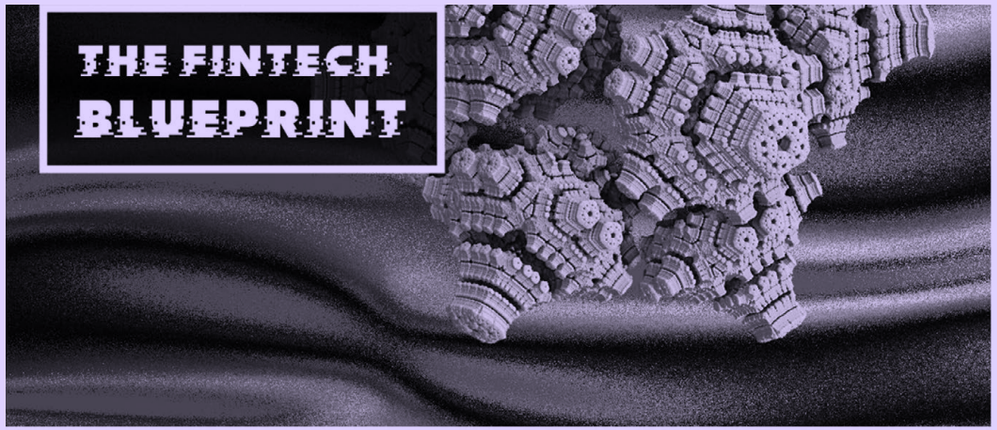




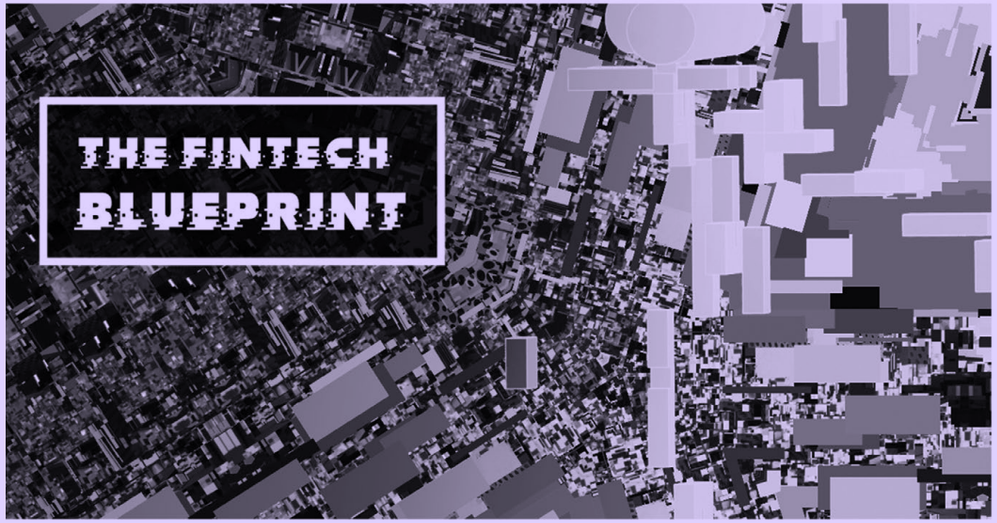

// close to the machine

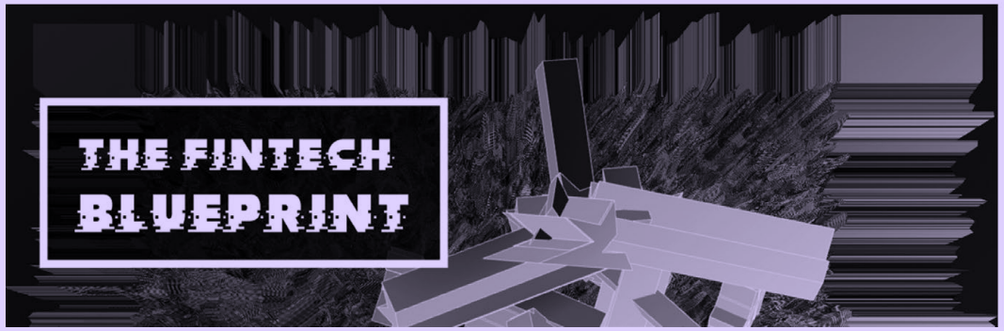




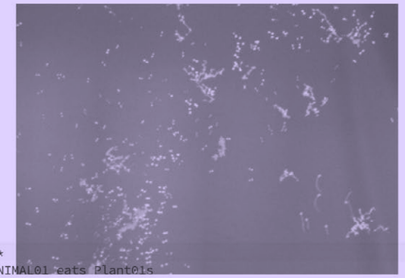



Processing powers a collage based brand system for my frontier tech newsletter with 100s of unique abstract covers and images generated by a deterministic system @lexsokolin



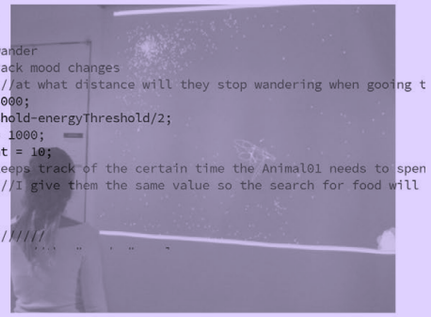





```
/*
ANIMAL01 eats Plant01s

add behaviors:
1.eat (what they do, but kill ressources)
2.hunger: need to eat, if not possible, die
3.Age
4.When eaten, look for a couple and reproduce
5.die and pass DNA
*/

//must get the plant from the control when the array was empty,
//from then on, ask a plant for the next one
```

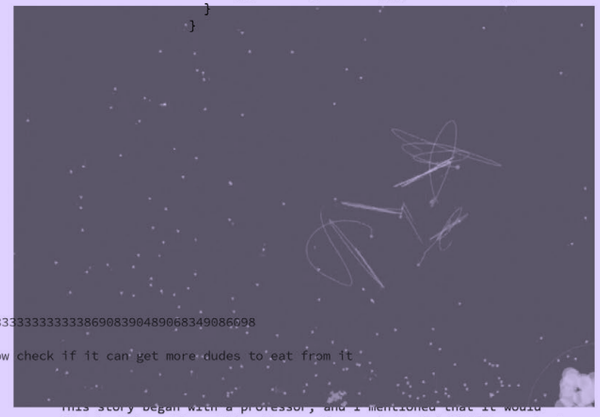
```
class Animal01{
  Vector3D loc, vel, acc, prevLoc;;
  color myColor;
  int sz;
  float theta;
  PathPlants myPather;//entity to communicate with, which controls the path
  //ArrayList pathArray;
  Vector3D myTarget;
  boolean gotTarget=false;
  int myId;
  int myMood=1;//0=eat/1=wander
  int myPrevMood=1;//to track mood changes
  int wanderForFoodVar=20;//at what distance will they stop wandering when gooing t
  int energyThreshold = 35000;
  int energy = energyThreshold-energyThreshold/2;
  int eatEnergyIncrement = 1000;
  int wanderEnergyDecrement = 10;
  int eatingCounter=50;//keeps track of the certain time the Animal01 needs to spen
  int maxEatingCounter=50;//I give them the same value so the search for food will
  Plant01 myPlantVictim;
  An01Mater myMater;
  ////////////////////////////
```



```
Animal01(int _x, int _y, color _c, PathPlants _p, int _sz, int _id, An01Mat
  loc = new Vector3D(_x,_y);
  prevLoc = loc.copy();
  vel = new Vector3D(0.0,0.0);
  acc = new Vector3D(0.0,0.0);
  myColor = _c;
  sz=_sz;
  theta = 0.0;
  myPather = _p;
  //pathArray = myPather.getPathList();
  myId=_id;
  maxspeed = 5.0f;
  maxforce = 0.5f;//was0.5
  wandertheta = 0.0f;
  //energy=50;
  myTarget = loc.copy();
  myMater = _myMater;
 }
```

```
public void getFirstPlant(){
  //look for next point to eat
  gotFinalPoint=false;
  myPlantVictim = myPather.gimmeAPlant();
  if(myPlantVictim!=null){
    Point p = myPlantVictim.getMeFoodPoint();
    //if(myPlantVictim.amIActive()){ //only if the plant i
      myTarget.setXYZ(p.x(),p.y(),0);
    //} else {
      //myMood=1;
    // }
  } else {
    if(myPather.gotAPlant())getFirstPlant();
    }
  }
```

```
public void checkPlantAvailability(float d){
  //      if(myPlantVictim==null){
    if(!myPlantVictim.amIActive()){
      eatingCounter=0;
      getNextPlant();//sdfgksdjoipjgsero0943333333333333333333333869083904890683490860
      //println("its dead");
    } else { //this means the plant is still alive, now check if it can get more dudes to eat from it
      if(myPlantVictim.canIEatYou()){
        // println(myPlantVictim.canIEatYou());
        if(d<wanderForFoodVar && !gotFinalPoint){
          Point p = myPlantVictim.gimmeAPoint(myId);
          myTarget.setXYZ(p.x(),p.y(),0);
          //yes it can, now if distance is smaller than 1, w
          gotFinalPoint=true;
        }
      } else {
        eatingCounter=0;
        getNextPlant();
      }
    }
  }
```

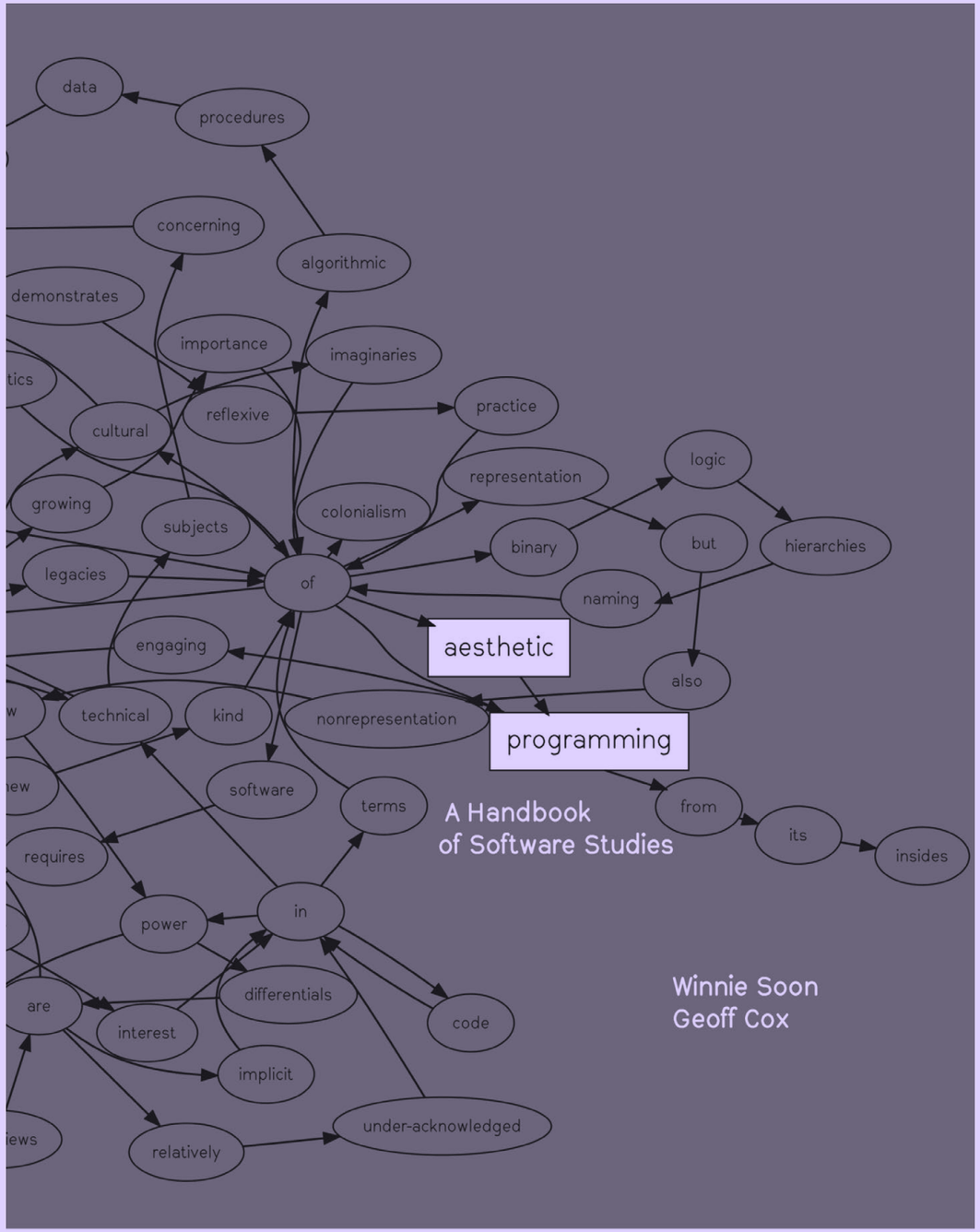


This story began with a professor, and I mentioned that it would finish with another. This other professor is Daniel Shiffman, who I had the privilege of being one of his first students. Thanks to his enormous patience, he was able to untie the first knots of my programming Ariadna strings, which were made up of functions and variables, and made up some fabulous classes that inspired the last project that is described here: *Digital Babylon*. Throughout the entire process, he constituted the most valuable support.

*Digital Babylon* is an interactive installation that is very different to the previous ones. In the first place, its aesthetic, which abandons work with real images in order to concentrate on synthesised ones. Synthetic and simplistic images, as the elements that make them up are intentionally simple so that movements and interactions can give the piece its intended feeling.

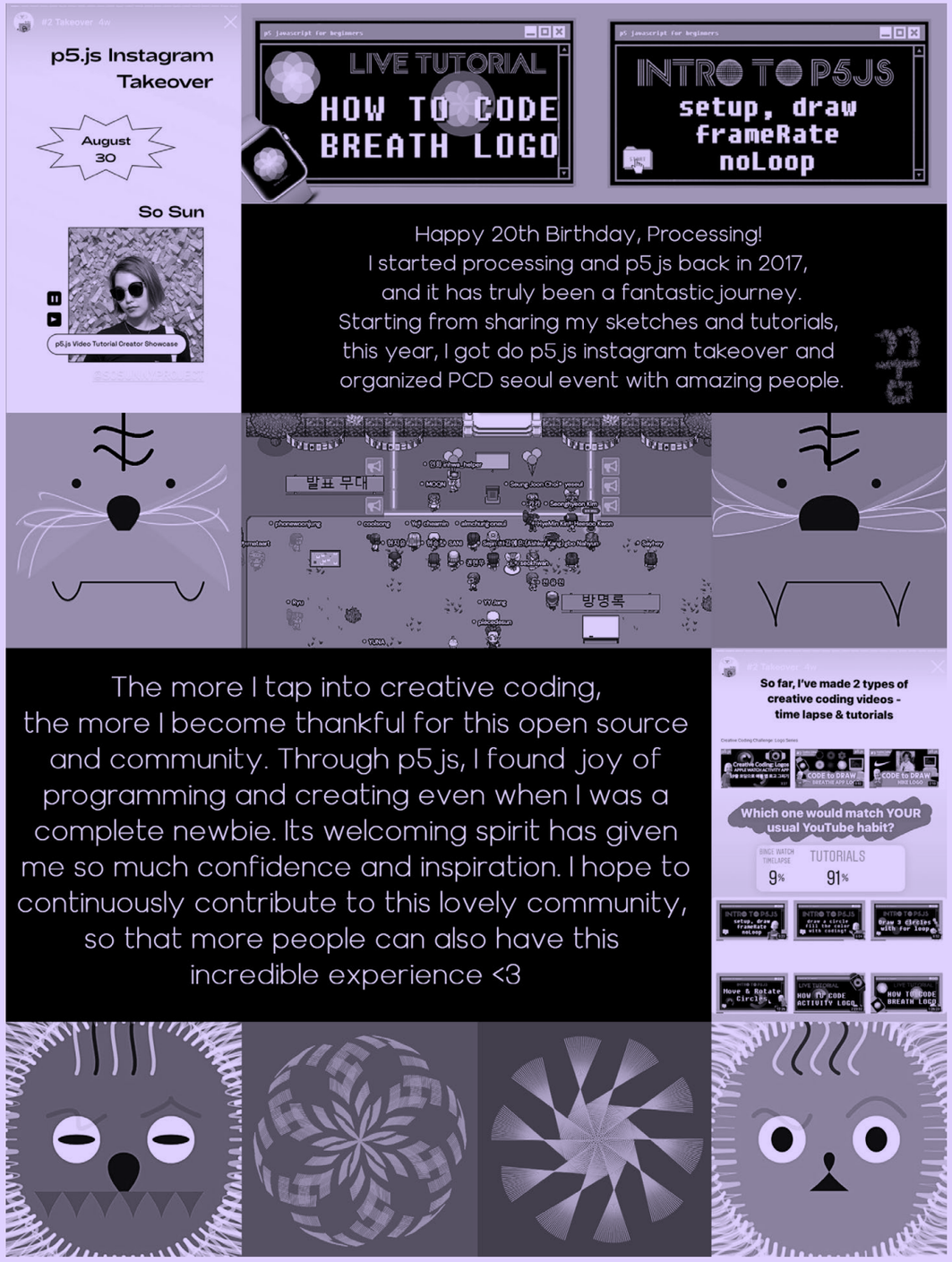
The piece presents the viewer with a small virtual ecosystem. The system works without the need for the viewer's intervention, although she can actually intervene in order to alter it, by entering a

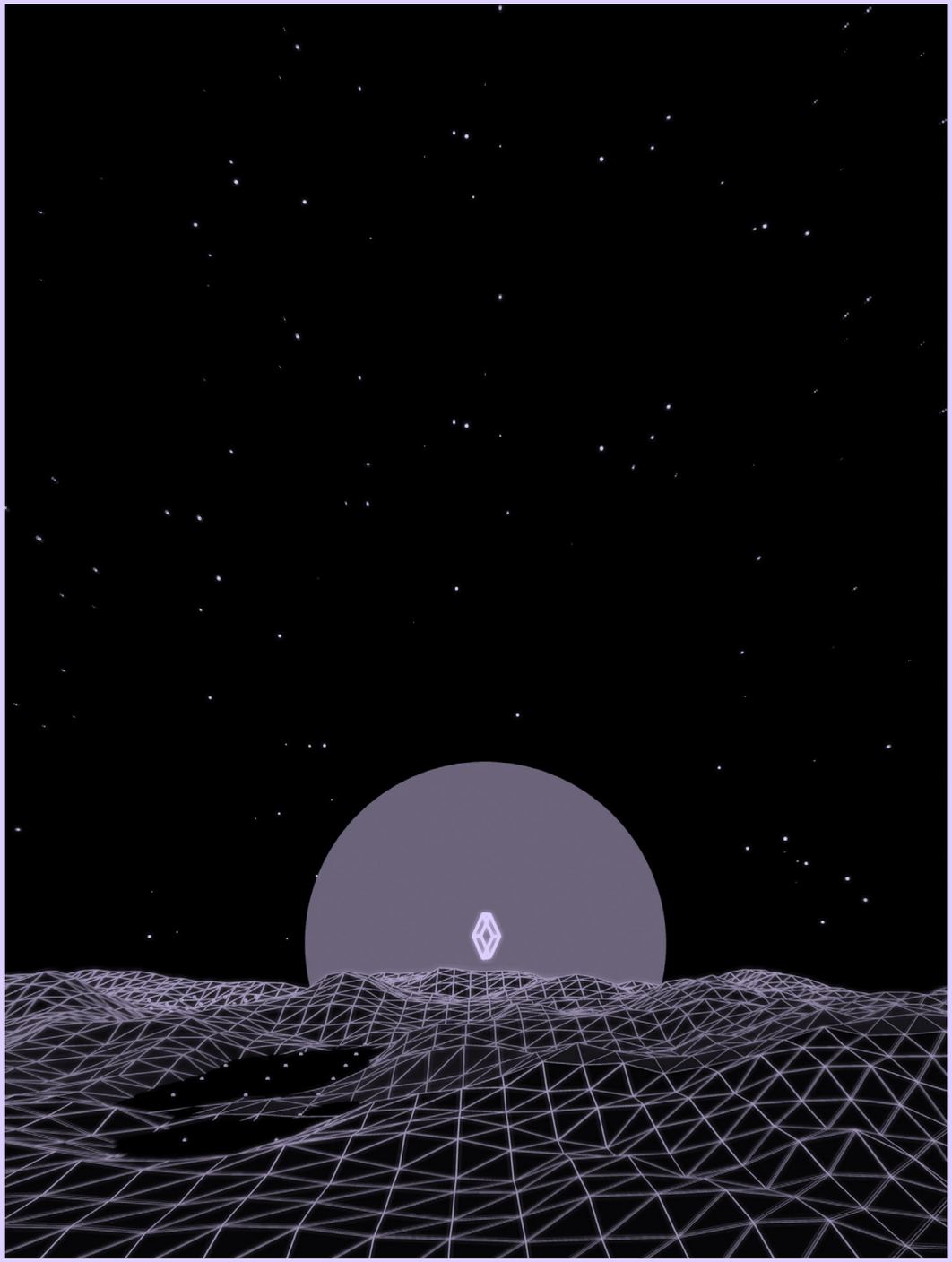





www.aesthetic-programming.net

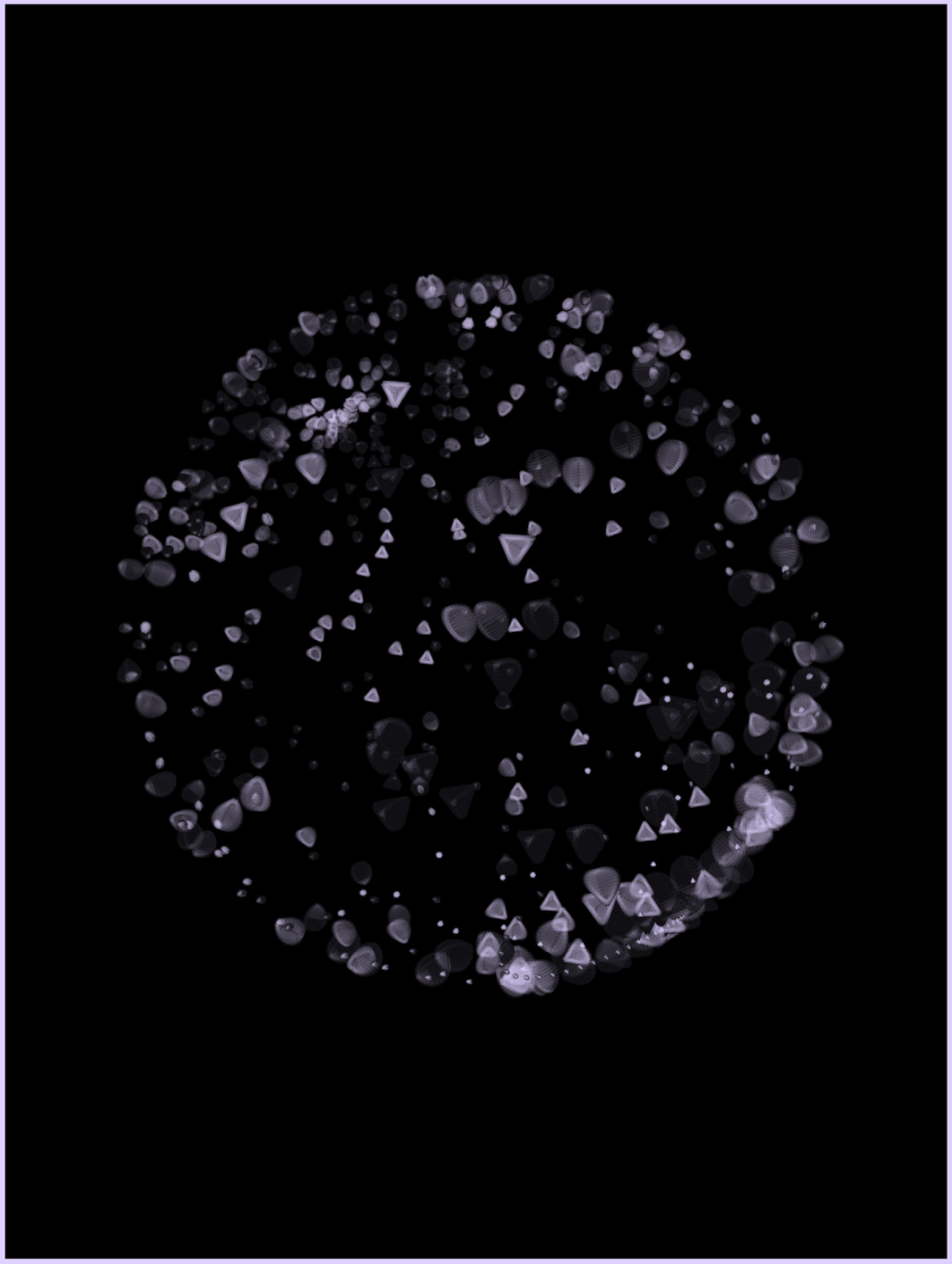



p5.js Instagram Takeover
August 30
So Sun
p5.js Video Tutorial Creator Showcase
LIVE TUTORIAL
HOW TO CODE BREATH LOGO
INTRO TO P5JS
setup, draw
frameRate
noLoop
Happy 20th Birthday, Processing!
I started processing and p5.js back in 2017,
and it has truly been a fantastic journey.
Starting from sharing my sketches and tutorials,
this year, I got do p5.js instagram takeover and
organized PCD seoul event with amazing people.
발표 무대
방명록
The more I tap into creative coding,
the more I become thankful for this open source
and community. Through p5.js, I found joy of
programming and creating even when I was a
complete newbie. Its welcoming spirit has given
me so much confidence and inspiration. I hope to
continuously contribute to this lovely community,
so that more people can also have this
incredible experience <3
So far, I've made 2 types of creative coding videos - time lapse & tutorials
Which one would match YOUR usual YouTube habit?
BINGE WATCH TIMELAPSE
9%
TUTORIALS
91%

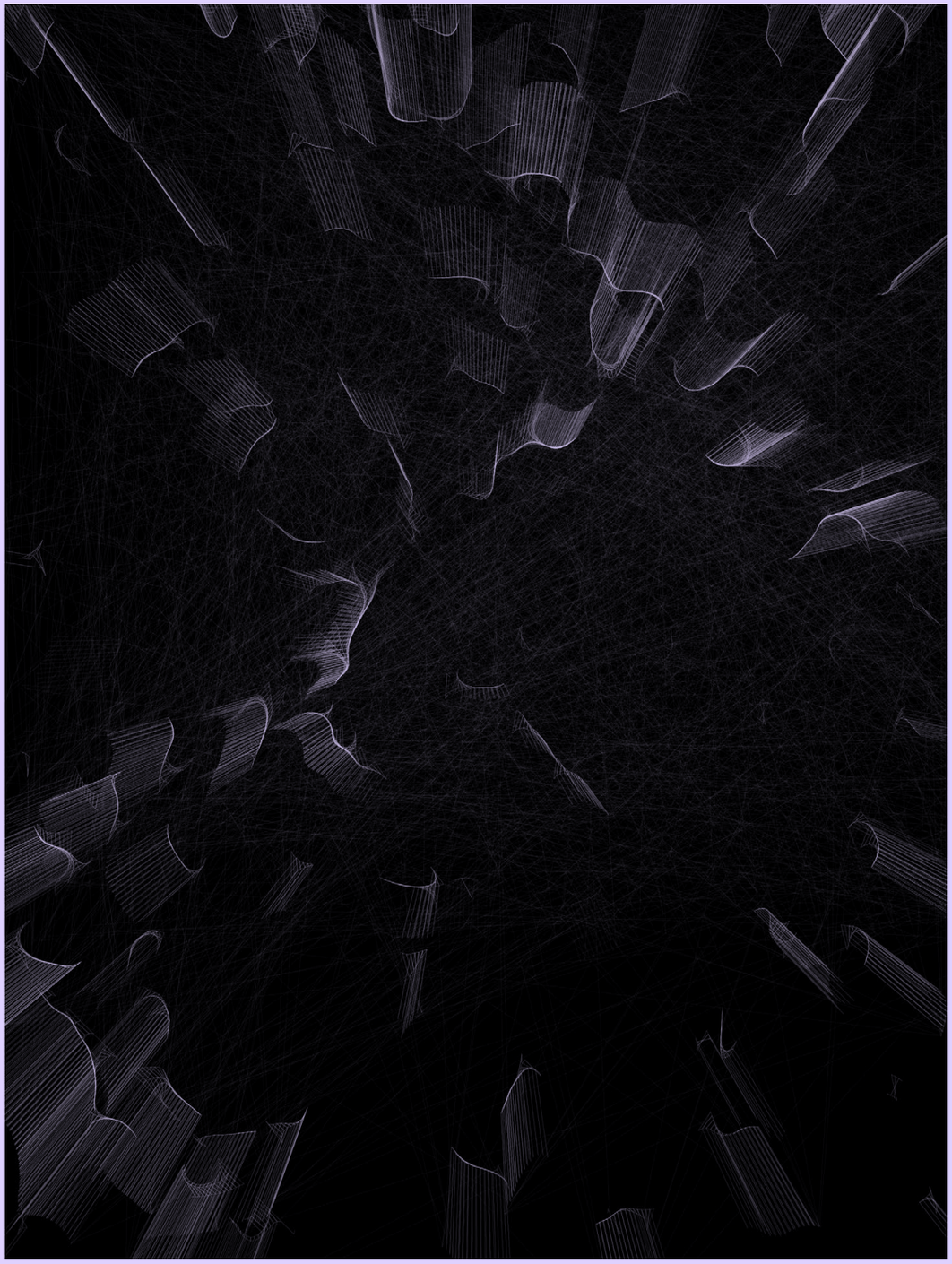





Pistols Choreo (I)
Sc
@SpaghettiCoder77



BARRY SPENCER @SPECULATYPE CON 517 832



@SPOONRIDER CON 518 833

**Tangled**

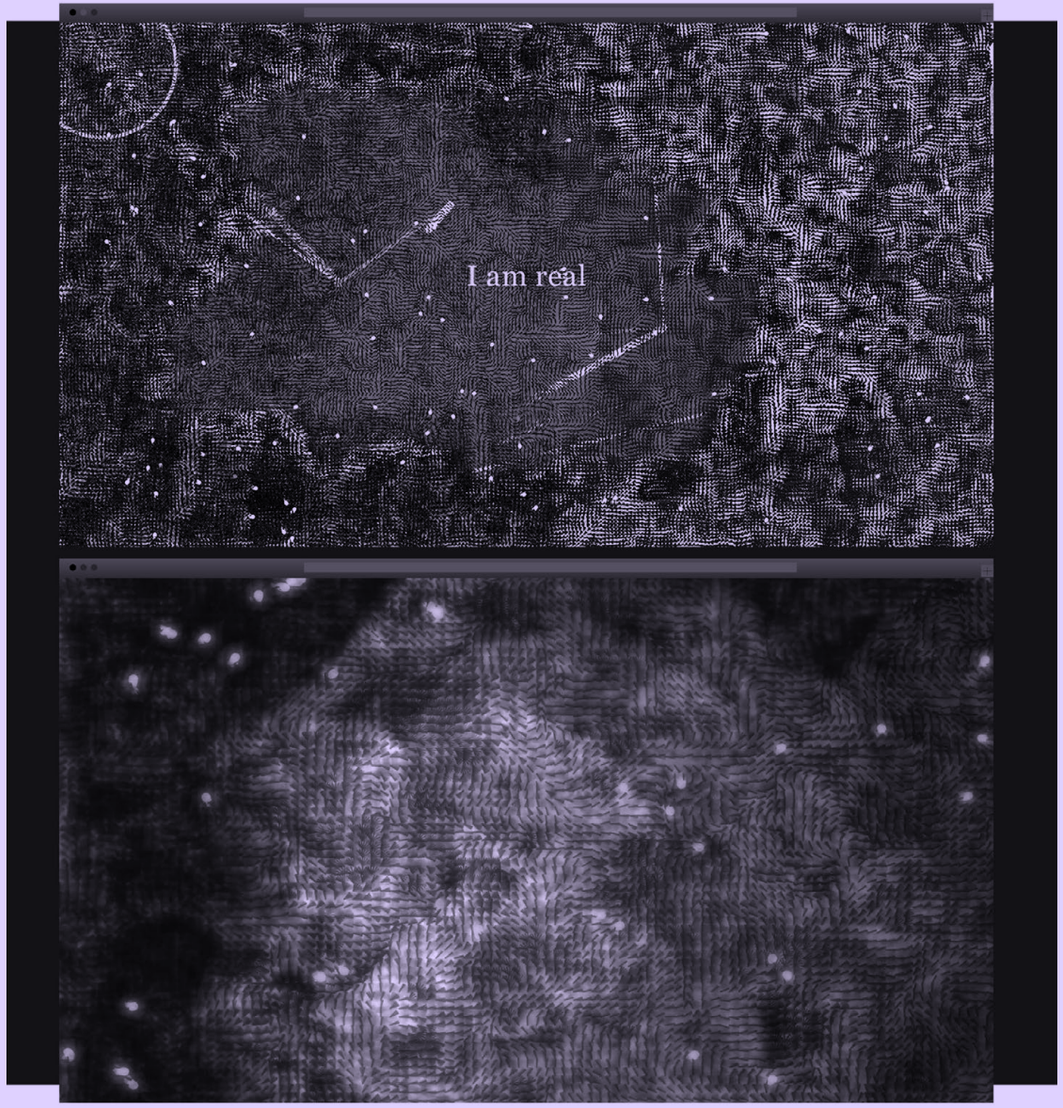
Generative 3D form using the HE_Mesh library and a custom particle system driven by Perlin noise in Processing. Rendered using external software.



**pink-noise.info, I am real 2020**



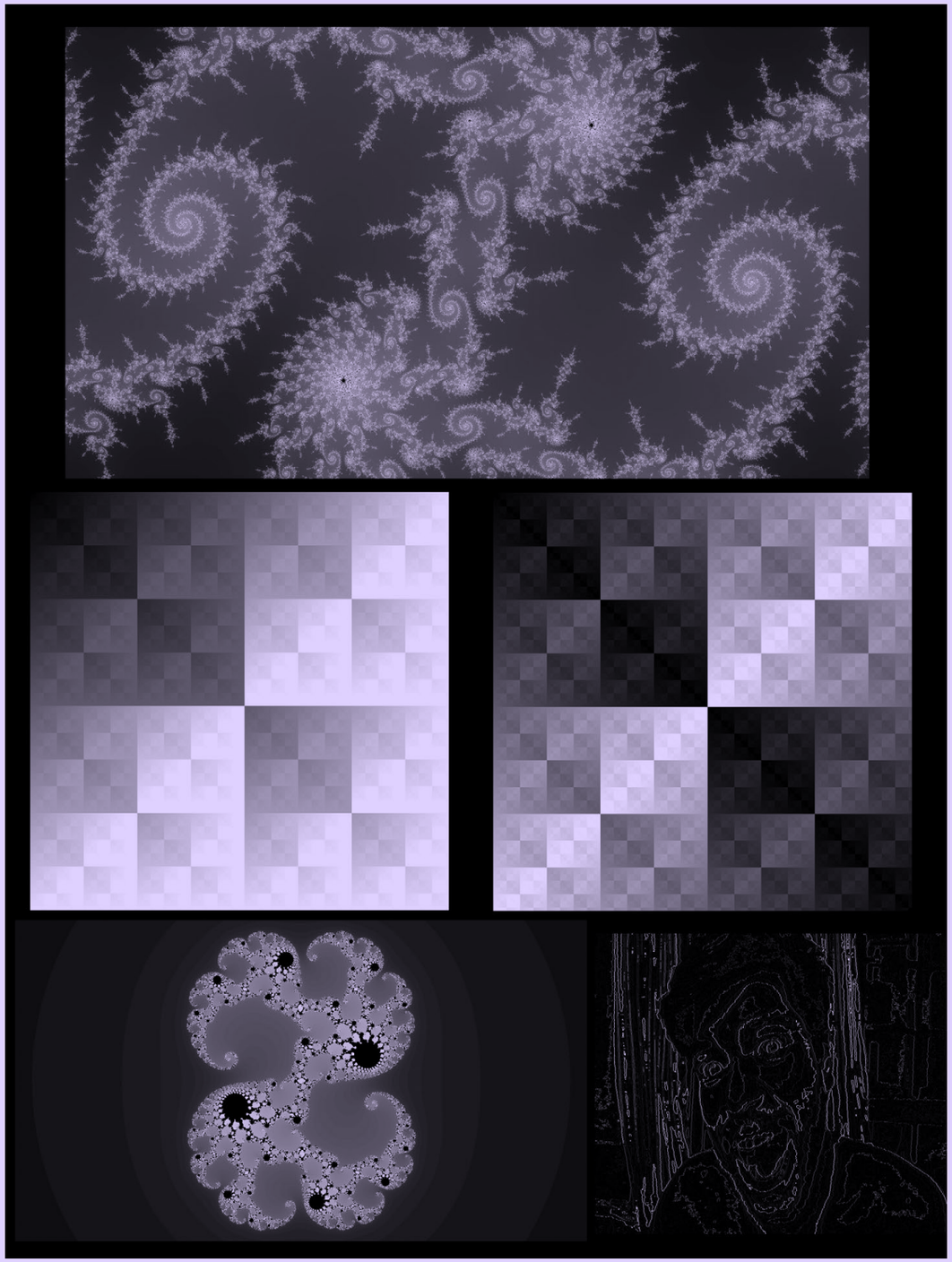
software, interactive site, video, original audio
duration: 2 minutes 10 seconds



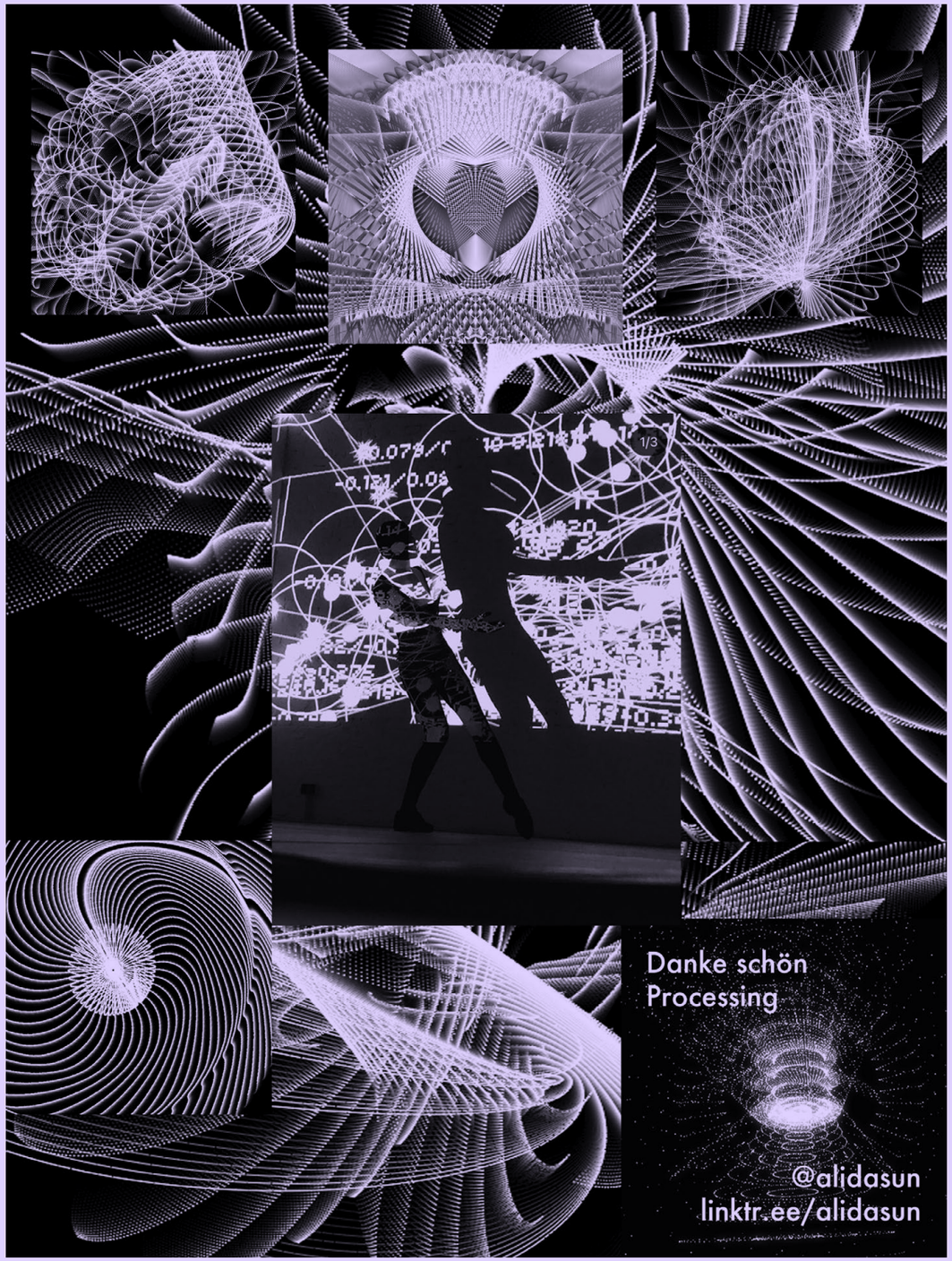

**Hurricane 2019**

software, digital image projection, original audio
duration: 1 hour (fifteen 4 min intervals)

made using p5.js

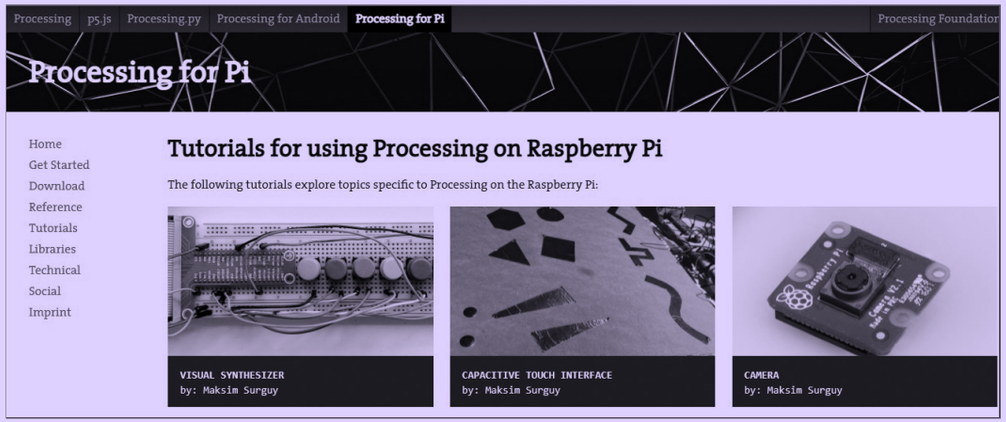




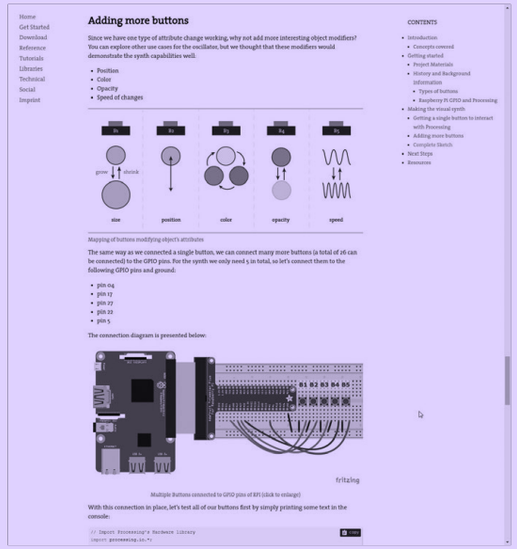

Danke schön
Processing
@alidasun
linktr.ee/alidasun




Description: Website and tutorials about using Processing on Raspberry Pi
URL: pi.processing.org
Author: Maks Surguy (@msurguy)




Caption: List of tutorials




Caption: Example from a tutorial


Caption: Example from a tutorial

During Google Summer of Code 2018, under mentorship of Gottfried Haider, Maks created a responsive open source website with reference and tutorials about using Processing on Raspberry Pi.

In these tutorials, readers can learn how to interact with sensors, cameras, buttons and other physical components to create interactive Processing sketches.


VISA
//
//  Play.P5.js
//  declare sprites
//
let tierra; // sprite, will need 2 images each 448 x 448
let carlosmoreno; // sprite, player character, maximum of 9
images
let cadaver; // sprite will need 1 mexico side
let gato1; // sprite, will need 2 on US side
let gato2; // sprite, will need 2 on US side
let waterLog; // sprite, flotsam & jetsam will need 1 top
edge touches border from Mexico side
let llanta; // sprite will need 1 top edge touches bank on US
side
let migraMan2; // sprite will need 3 [tom, dick, harry,]
walking on banks
let migraMan1; // sprite
let migraSUV; // sprite, will need 1 on center of lane [
gto, exp] currently have wrong art. facing wrong direct
may not have art
let migraHelo1; // sprite, will need 2 [huey, bell,
let visa; // sprite, goal
let migraHelo2; // sprite
let migraMan3; // sprite
let laMigra; // group, it may be better to call t s
opponents or hazards because they include flotsam



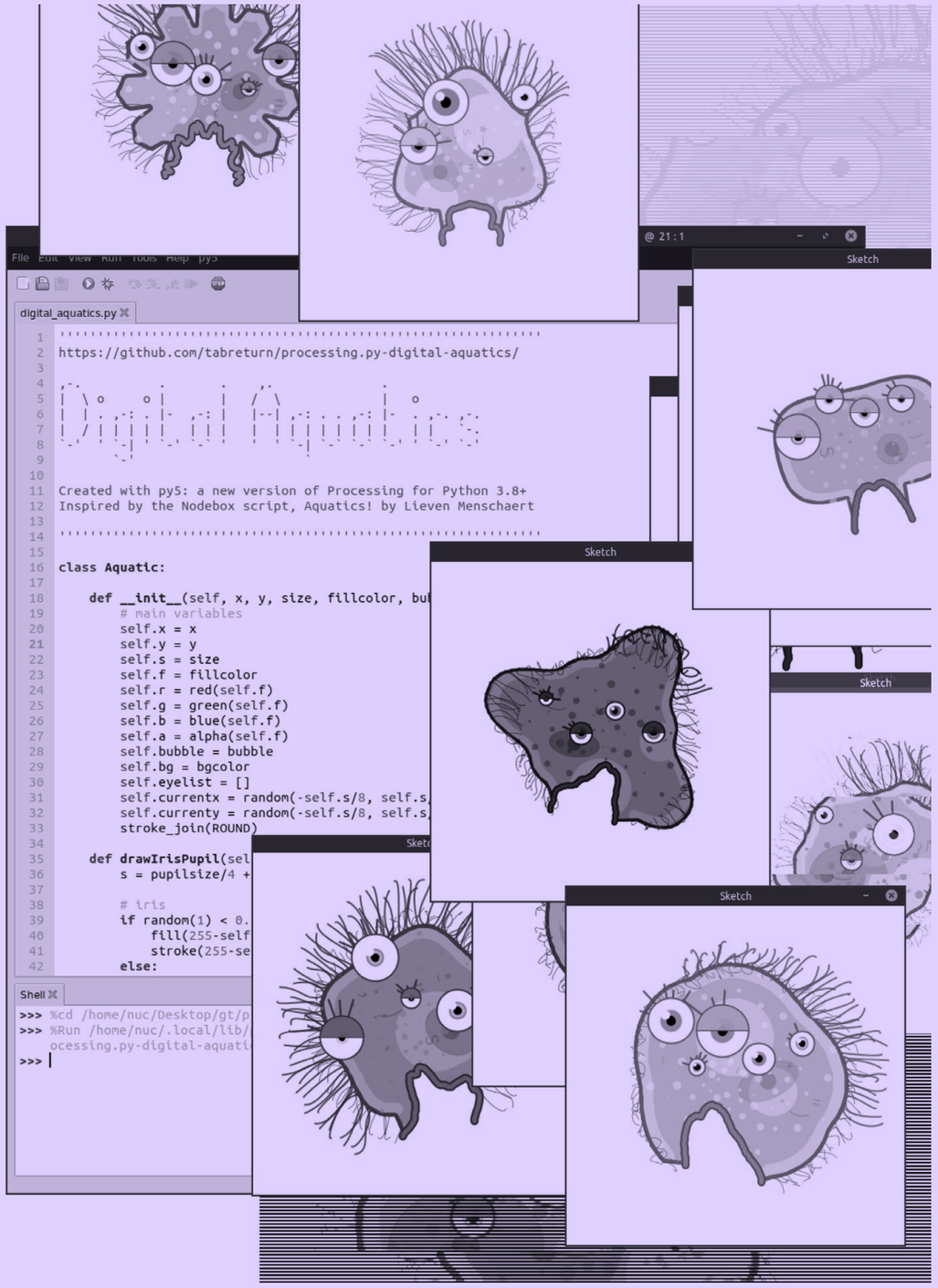

ADULTING SHOULD BE OPTIONAL
#ADULTING
SOCIAL-MEDIA LIVE DATA, IMMERSIVE AND INTERACTIVE ART INSTALLATION
TUANGSTUDIO.COM



MY STUDENTS
IN SAN MARINO
@UNIRSM x DESIGN
SINCE 2015

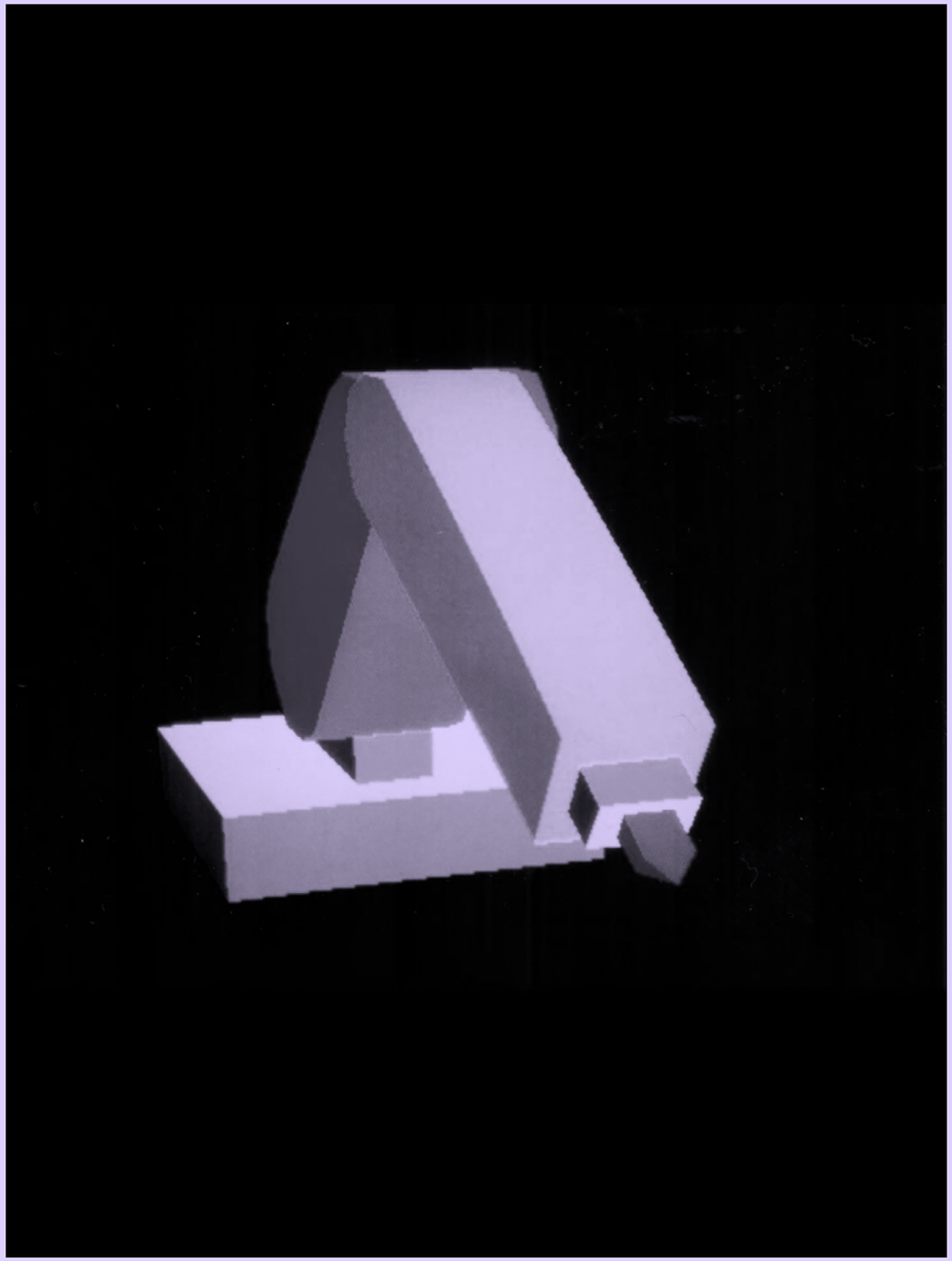
@FUPETE 21

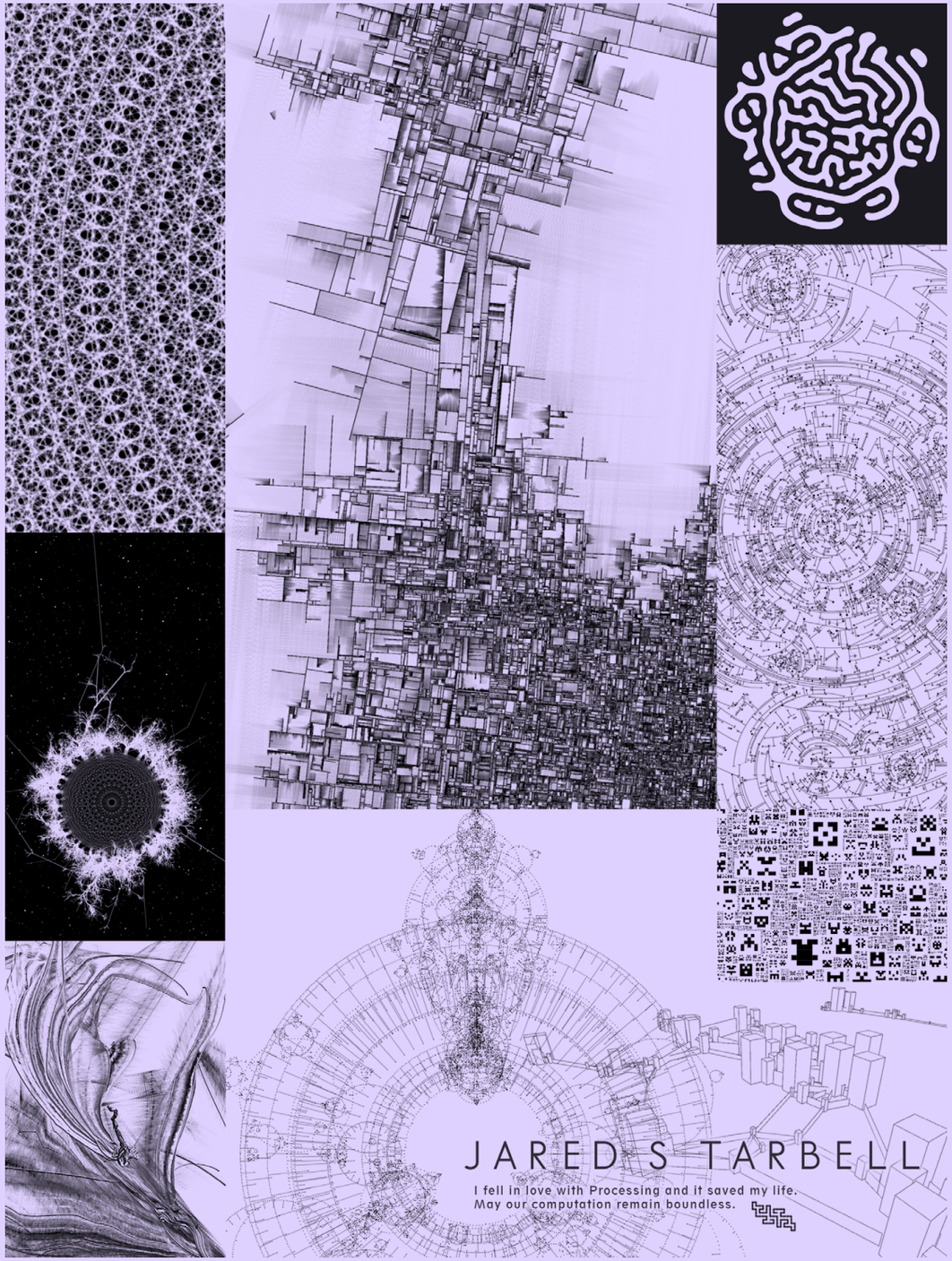





Processing
臺灣國際社群日
Processing
Community Day,
Taiwan
C-LAB
未來媒體藝術節
FUTURE
MEDIA
ARTS
FESTIVAL
2021.10.19-24
臺灣當代文化實驗場 C-LAB
No. 177, Section 1, Jianguo South Road,
Daan District, Taipei City, Taiwan (ROC)
指導單位 Supervisor
主辦單位 Organizer










## Shader Mode for Processing Development Environment

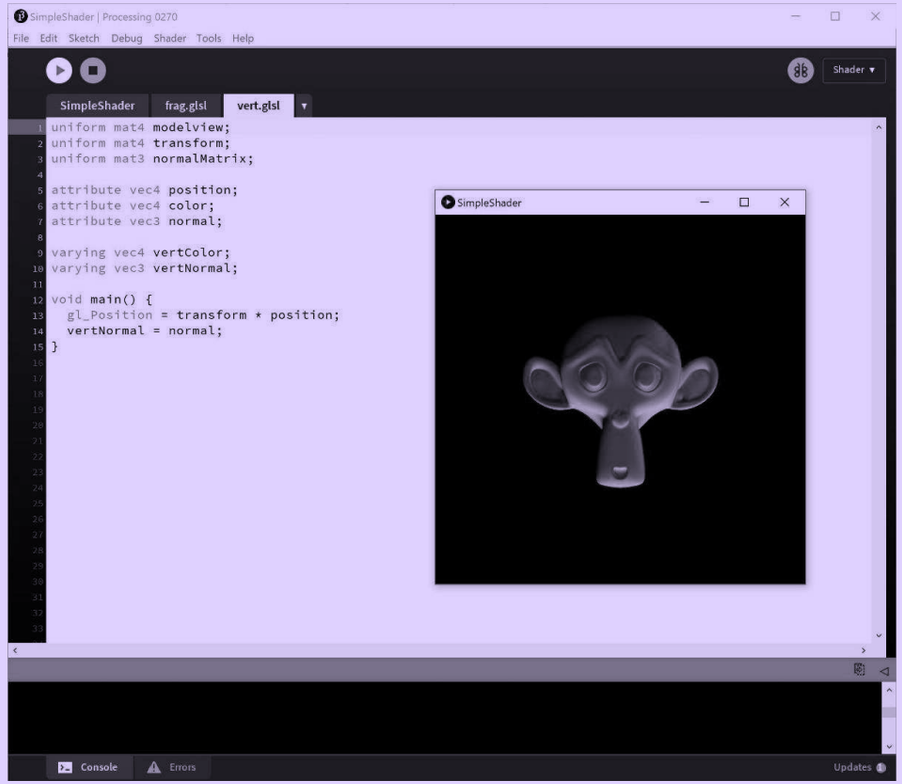


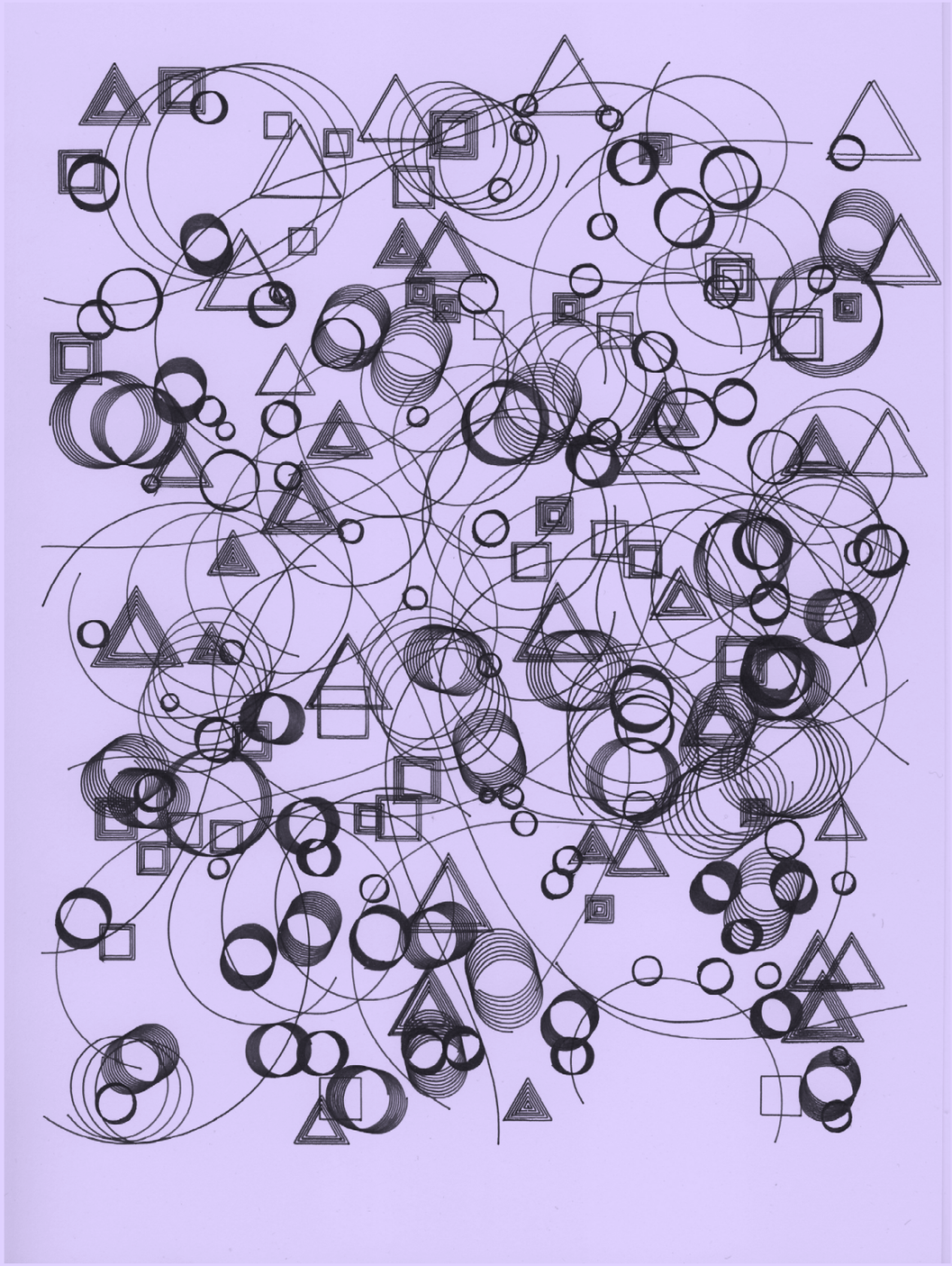

*Figure shows a shader program running in the Shader Mode tool in PDE with multiple shader files (frag.glsl and vert.glsl) opened in separate tabs. The smaller sub-window shows a rendered model of the Blender monkey head "Suzanne".*

The development of Shader Mode was inspired by a project Izza Tariq did for Google Summer of Code in summer 2018 and was later continued as part of their master's thesis. The motivation behind developing this tool was to facilitate the use of shaders in Processing IDE and help bridge the gap between the creative ability of computational artists and the technical expertise expected of them.

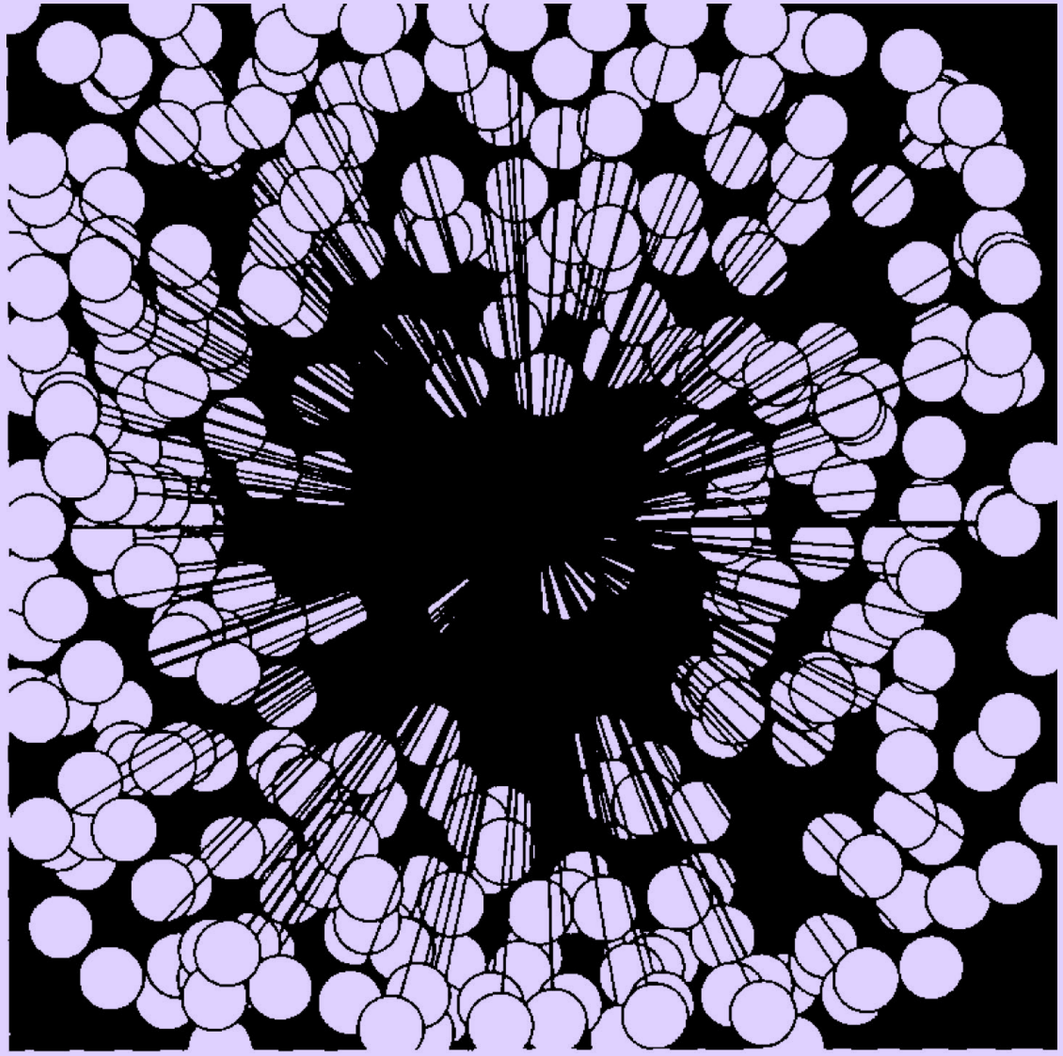
The concept of a Shader Mode was based on the idea of modes in Processing such as Java, Python, Android and p5.js (JavaScript) that allow flexibility with different types of platforms and programming techniques. Shader Mode is supported on Processing versions 3.5.x and higher.

Below are different features and functionalities Shader Mode offers:

- Create and save new shader and sketch files
- Open and edit multiple shader and sketch files side by side
- Syntax highlighting of GLSL code to provide better code readability
- A shader menu for additional resources such as tutorials and shader templates
- Shader templates that include minimal shader code to give users a starting point to develop different types of shaders
- A set of basic and advanced shader examples for inspiration

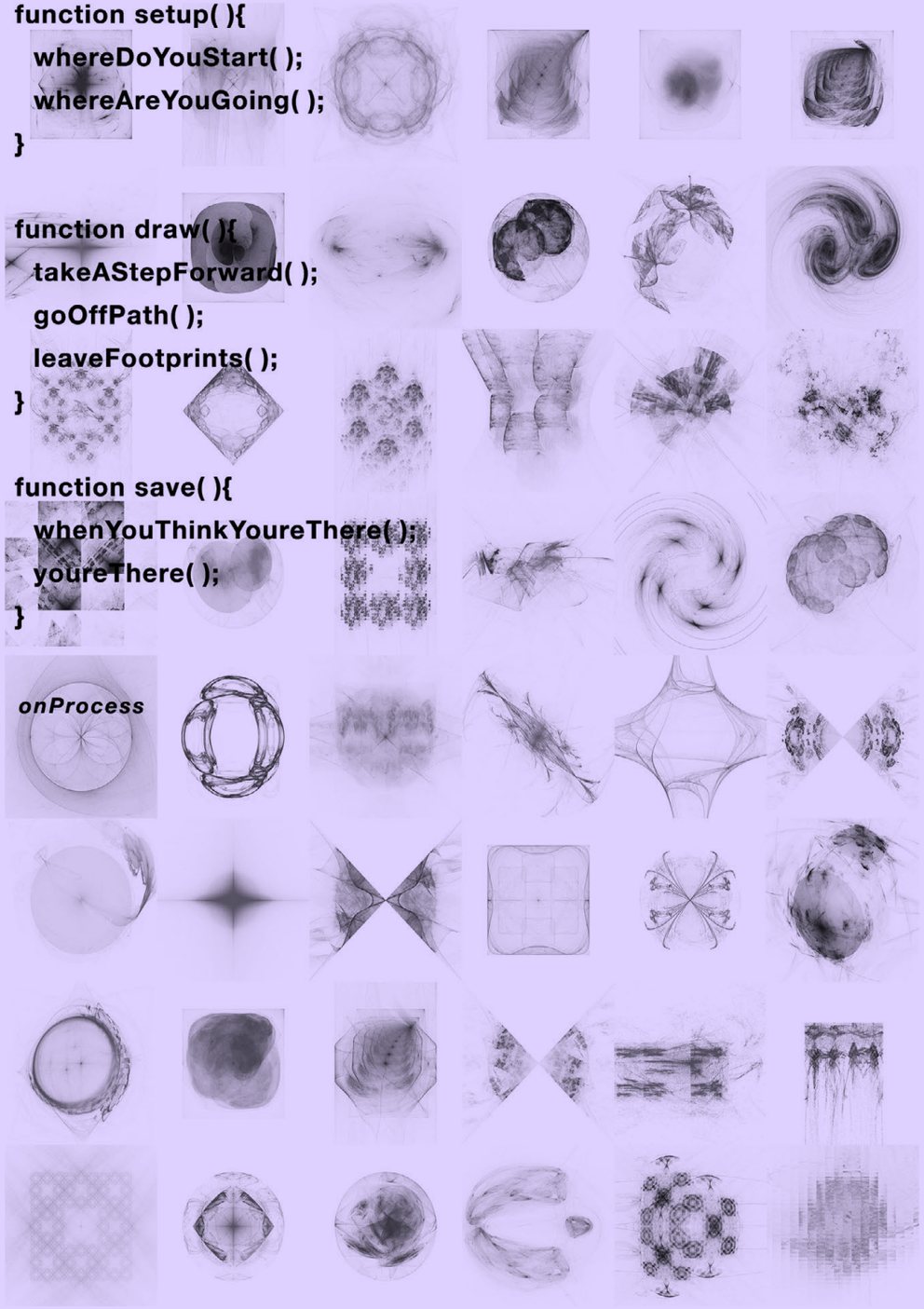




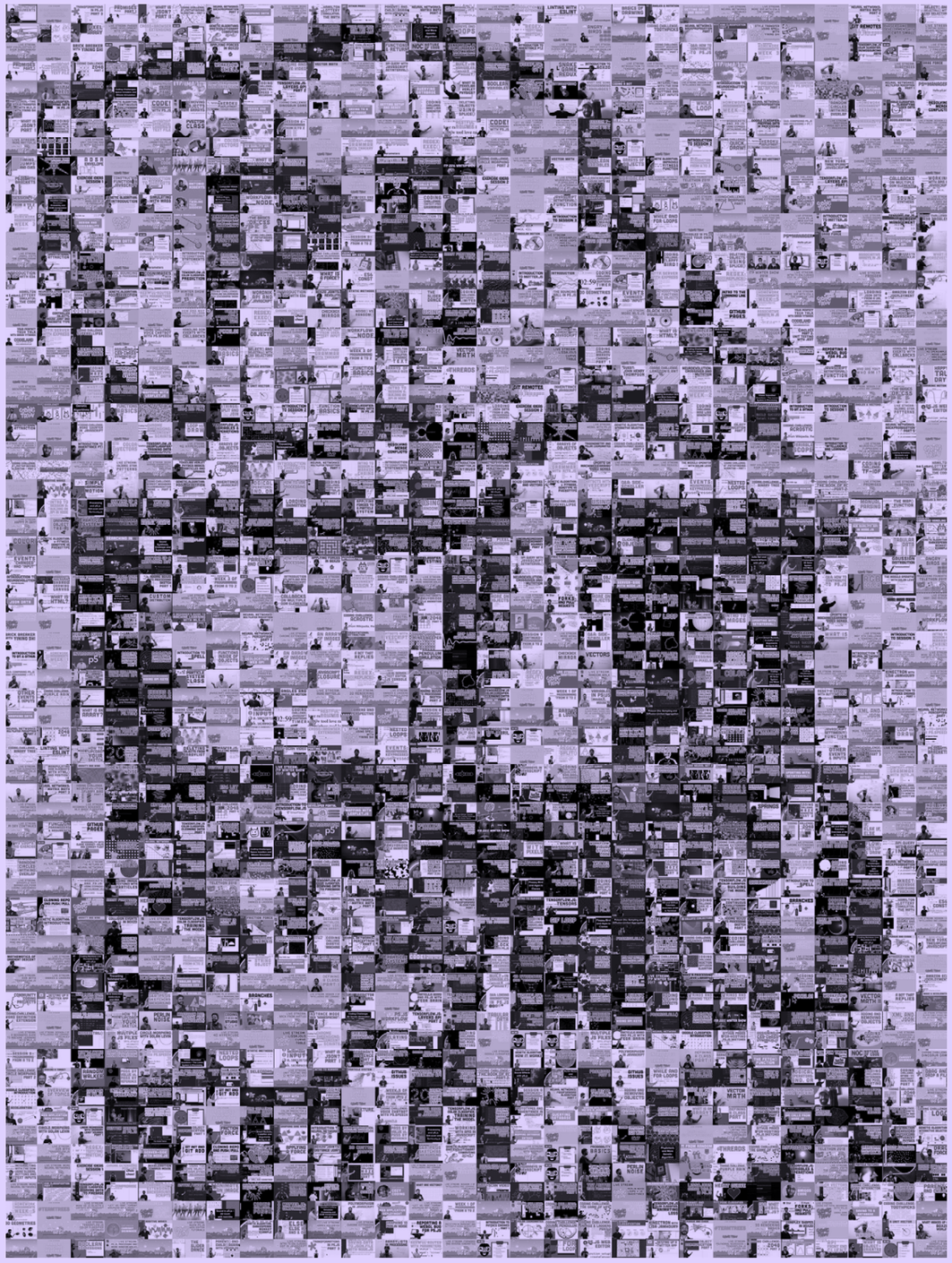


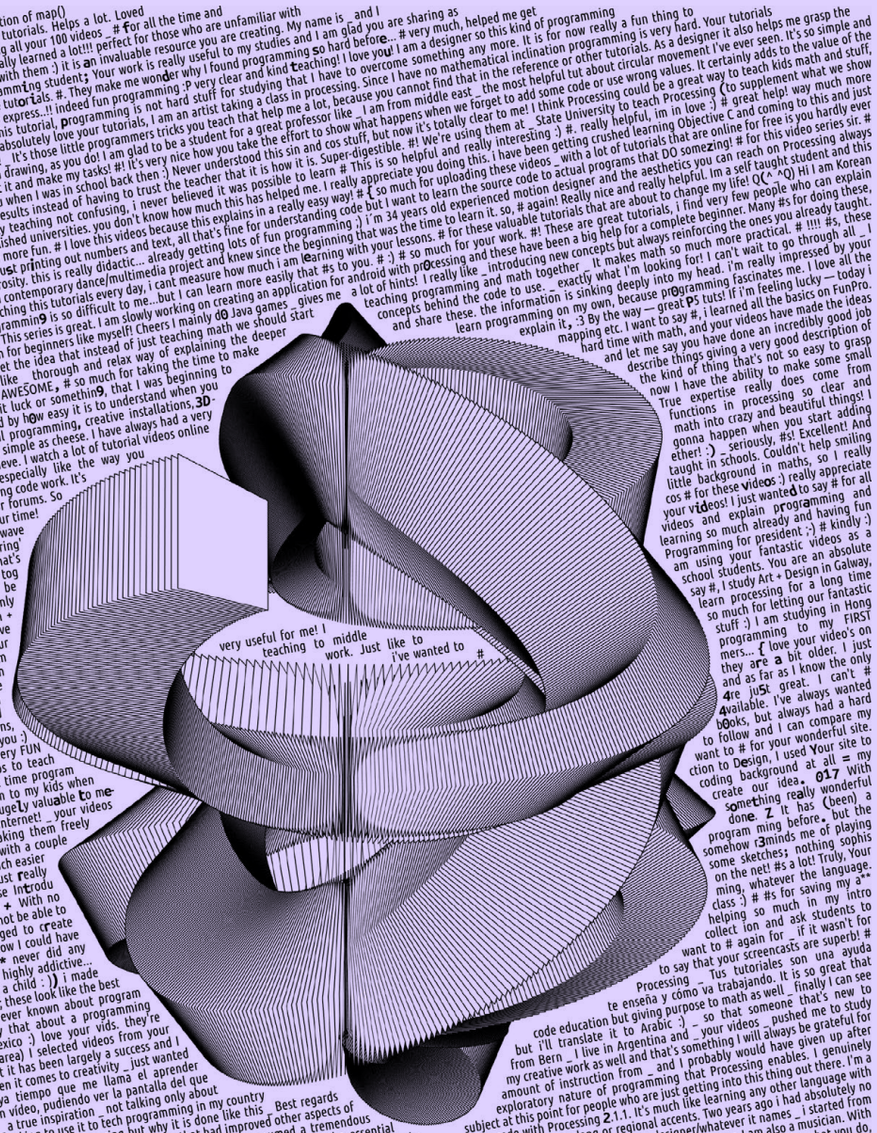
```
function setup( ){
  whereDoYouStart( );
  whereAreYouGoing( );
}

function draw( ){
  takeAStepForward( );
  goOffPath( );
  leaveFootprints( );
}

function save( ){
  whenYouThinkYoureThere( );
  youreThere( );
}
```

*onProcess*





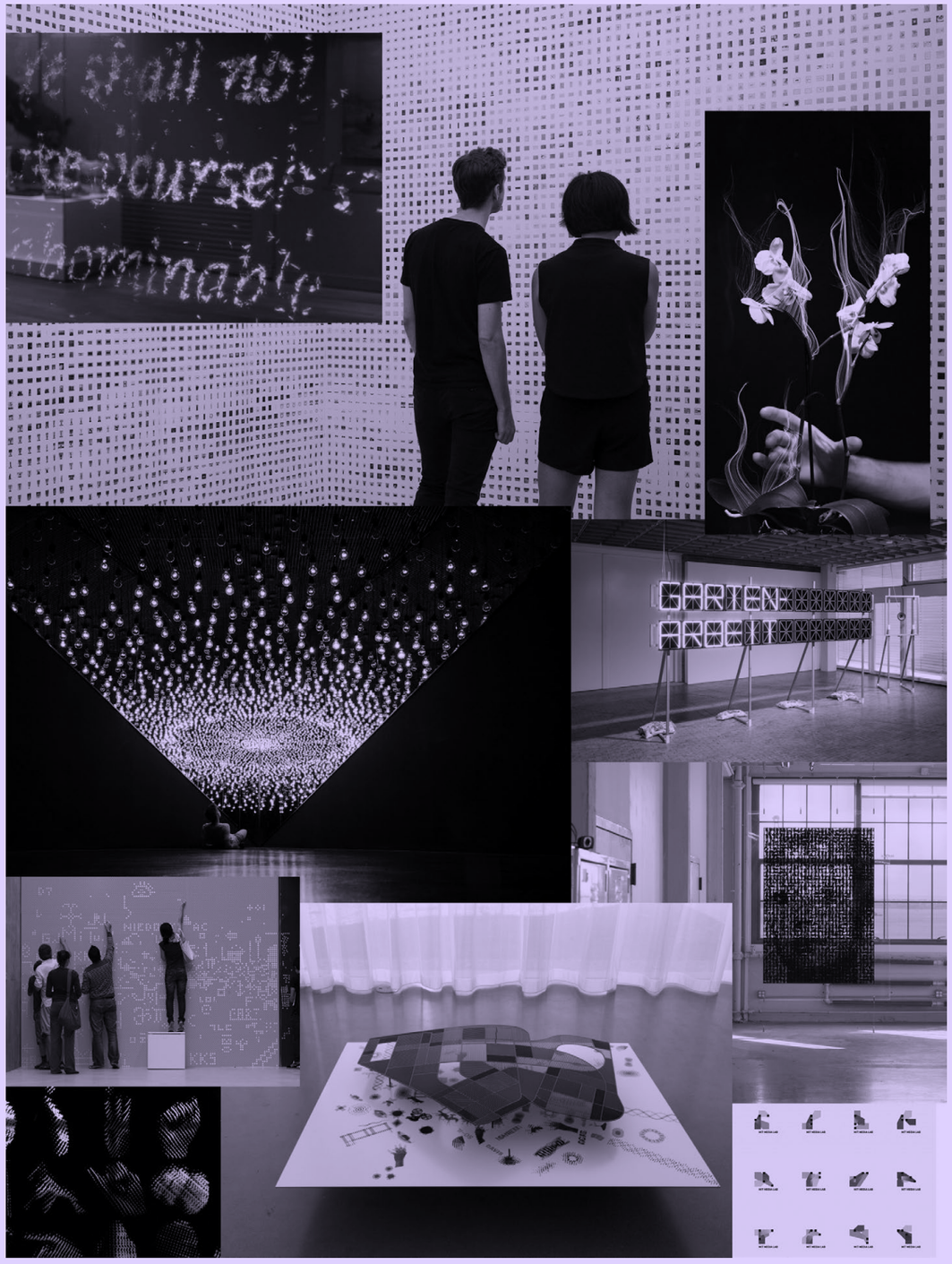



very useful for me! I
teaching to middle
work. Just like to
i've wanted to #



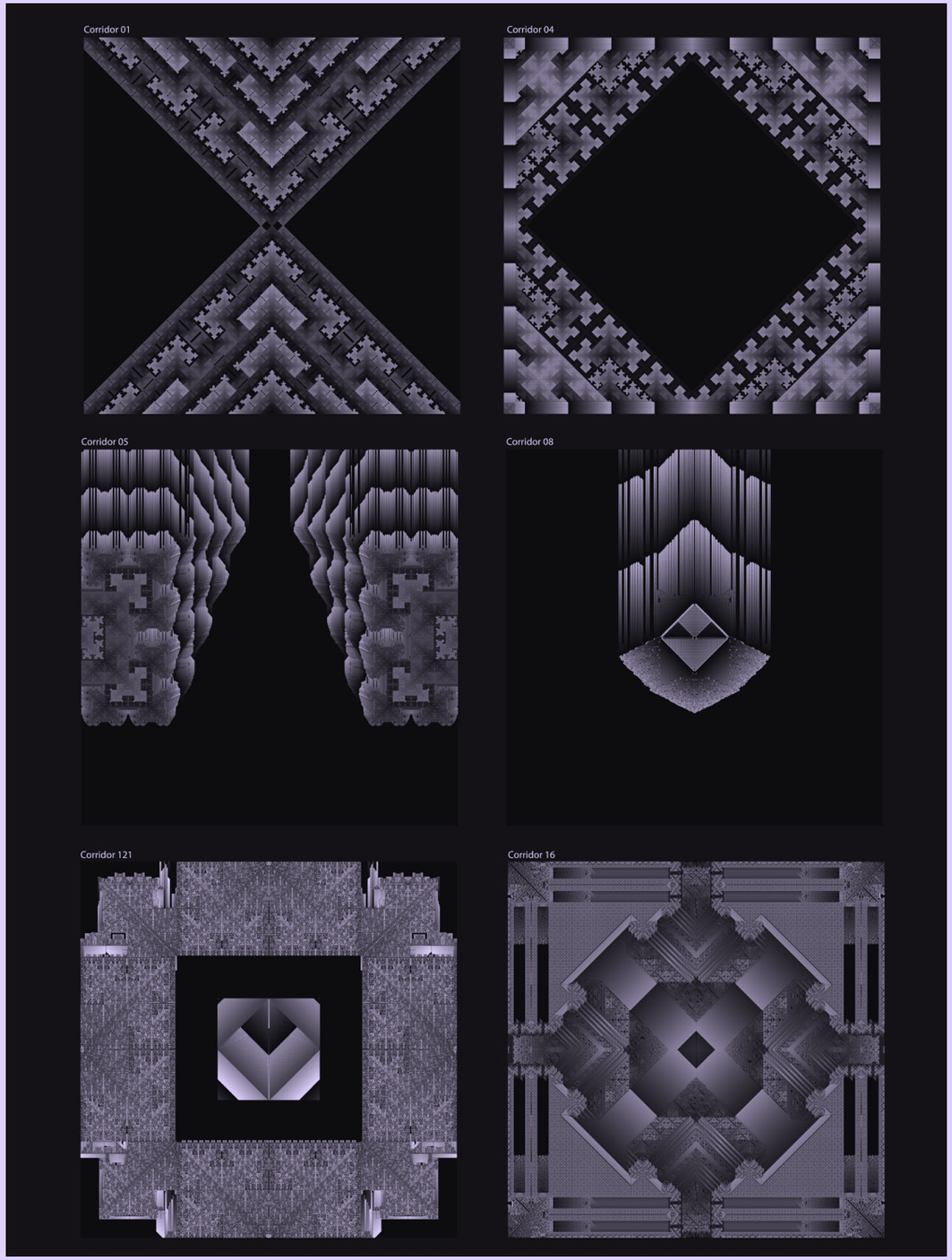

In 2011 I was asked to come up with a new visual identity for the school I was currently graduating from, the MIT Media Lab. I attended the lab because I wanted to study in the "Physical Language Workshop", formerly known as "Aesthetics and Computation Group", run by John Maeda. Once we started the project it quickly became obvious that we want to be inspired by the playful and manifold Media Lab business cards created by the amazing Muriel Cooper and that this visual identity could possibly become an algorithmic, generative identity. It would be inspired by the community of the Media Lab: Creative people from all kinds of backgrounds coming together, inspiring each other and creating something new. One logo that could move, be flexible and change, yet could be claimed by each individual lab member, whether student, faculty or staff. Given the legacy of Cooper and Maeda it was only fitting that we would use Processing, created by alumni Ben Fry and Casey Reas, to create this generative visual identity. I will always be grateful what Processing, and the creation of this project using Processing has taught me along the way. —Richard The



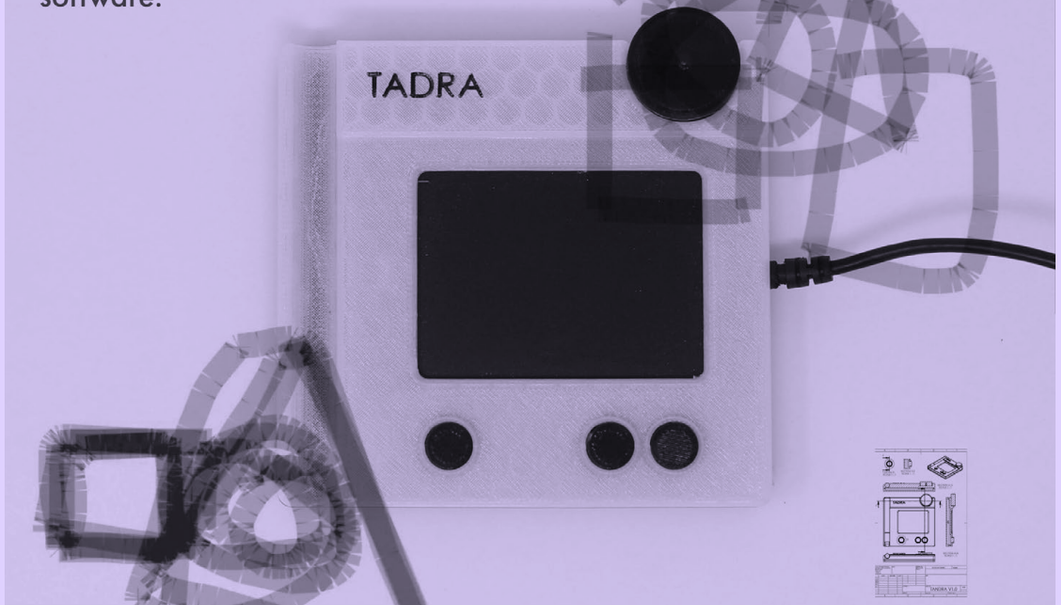



THANK 4
THE CODE DREAM




Corridor 01
Corridor 04
Corridor 05
Corridor 08
Corridor 121
Corridor 16

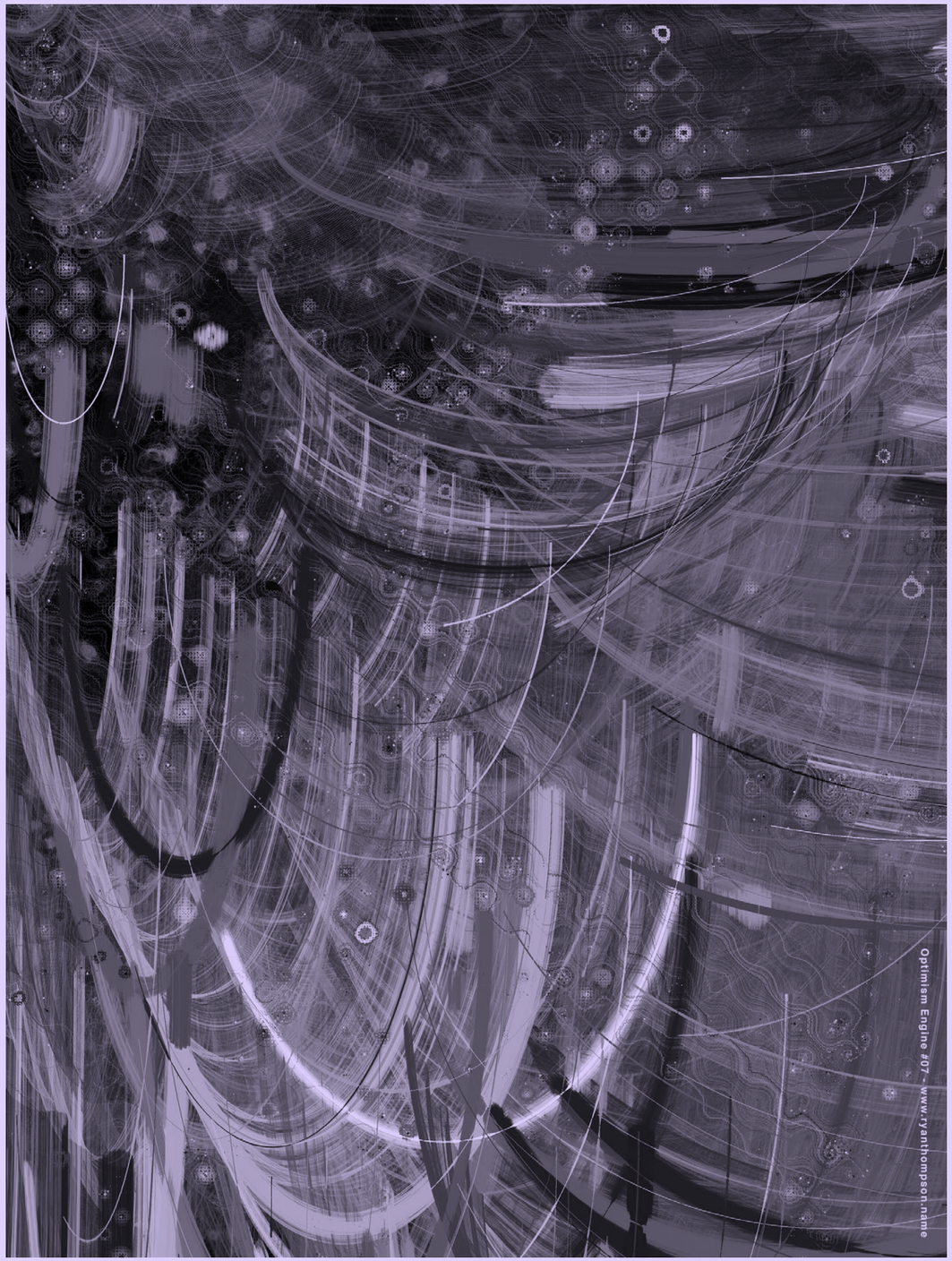
**TADRA** is a sonically guided drawing experience created with Processing software.




It explores the relationship between drawing, communication, shape and sound. As participants draw geometric primitives to communicate to others, audio frequency and amplitude changes occur along the X- and Y-axes.

Programming: Jesse C Thompson
Prototyping: Samuel Wangsaputra
demo: https://s01e01.xyz/tadra





# Proposal For A New Trans Flag

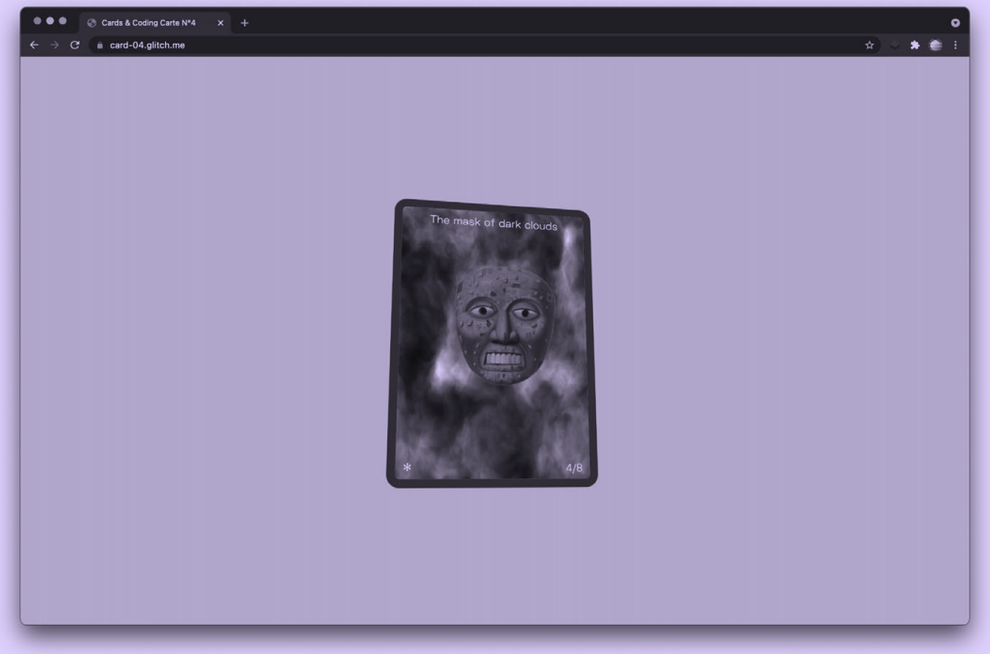
Chelsea Thompto
2020

A few years ago I had a dream of a trans flag whose colors shifted as endlessly as its form in the wind. A version that spoke both to multiplicity and to the underlying system trans folks traverse.

So, in mid 2020 I coded this 2D version, procedurally generated using JavaScript, this flag will infinitely change color in its horizontal bars. While the colors shift, the black and brown triangle portions stay the same, a gesture towards the need to center black and brown trans voices and experiences. The design draws from the original trans flag, the black trans flag by Raquel Willis, and the progress pride flag by Daniel Quasar.

https://cthompto.github.io/new-trans-flag/





WOODCUT FLOW BY JASON TING



NAVEL LA

THE ARRAY[]

AJITESH LOKHANDE &
HARSHALI PARALIKAR
BOMANI MCCLENDON
BRILEY LEWIS
DALENA TRAN
ELANA SASSON
ELIZABETH LIN &
POETECHS
EVERYWWWHERE
FLAWLESS HACKS
GONZALO MOIGUER
HANNAH FLYNN
HECTOR TORRES
KITT PEACOCK
LIBRENAUTA
MAPPING FEMINIST LA
MEGHNA DOLAKIA
NAHEE KIM
NITCHA TOTHONG &
KENGCHAKAJ KENGKARNKA
RIGOBERTO LARA GUZMÁN
ROBIN NETHERTON
SPRUCE LEE
XIN XIN
ZAINAB ALIYU

TABLERS

ALT.CTRL.
A.RUN
AMY BUREK
AMY WIBOWO
AMY ZHOU
ANTHONY TRAN
COLOR CODED COLLECTIVE
E. L. GUERRERO
FREE RADICALS
GHOST JAM
IMANI RITCHARDS &
CHANTELL WILLIAMS
JAMIE RENEE WILLIAMS
JASON ALDERMAN &
INNA ZILBER
JOYCE S. LEE
KEVIN PRINCE
LINHTROPY
MAYA FRIEDMAN
MIKE MATTHEWS &
JOJO LEOVONCHIONG
OLIVIA HARPER WILKINS /
AMAZINE SPACE
RACHEL JOY VICTOR
STEFANIE TAM
TECH WORKERS
COALITION, LA

08.11.19

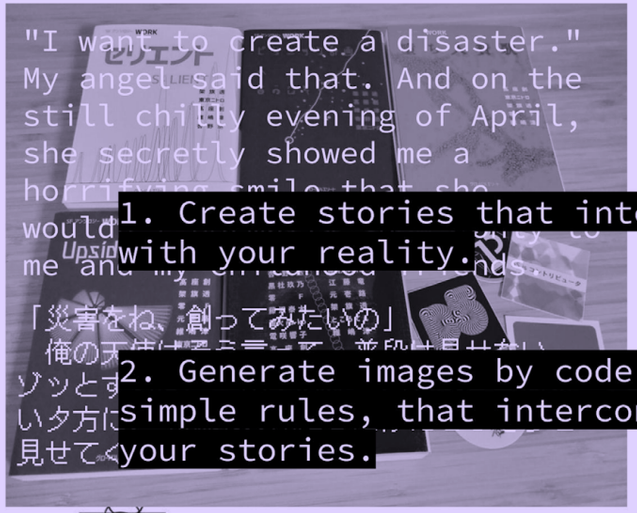
WORKSHOPS

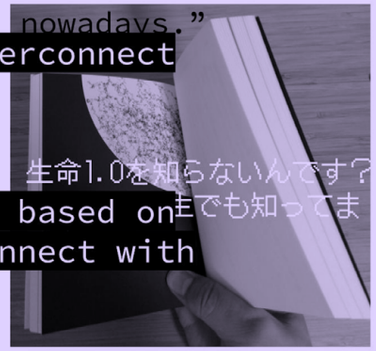
CASEY REAS
CRITICAL THEORY INDEX
JULES KRIS
RACHEL JOY VICTOR

TiNY™

TINY TECH ZINES

# グローバルエリート
# GLOBAL ELITE




1. Create stories that interconnect with your reality.

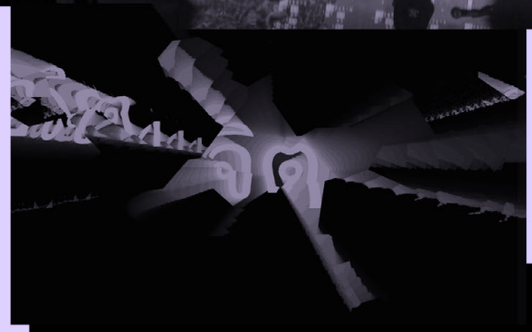
2. Generate images by code based on simple rules, that interconnect with your stories.

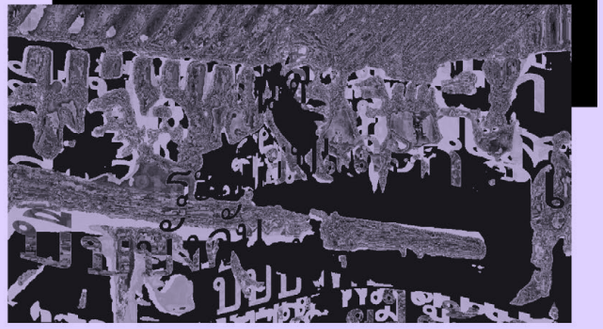
3. Understand EVERYTHING in your world interconnect with each other .
(And enjoy it!).

The station platform was
crowded with people in
black dust suits. The black
fabric suits was damp from
the
refle
appea
lined up in a row
gaps looks like a
insect. Takashi a
were sitting on the
in a sloppy fashion,
ignoring the cold gaze of
the suit-beetles.

ホームは真っ黒な防塵スー
ツの人々でごった返してい
る。霧で湿ったスーツの黒
生地が光を反射し、隙間
にそれらスーツが
る様子はある種の
えないこともな
シとノエルはスー
の冷たい視線やあ
まな舌打ちを無視し
てベンチにだらしない恰好
で居座っていた。







Uneven Geographies of Sound event at Wonderville, New York on Sep 15, the show is part of the CityArtist-Corps. Kengchakaj and I presented project Jitr.

Jitr( จิตร ) is a speculative, imaginary electronics ensemble using computer programing and live coding to generate and provide possibilities of decolonized Southeast Asia sound culture and its visual representation. Jitr( จิตร ) is interdisciplinary by its nature stretch beyond the art of sound making that covers improvisation, live coding, electronics, music technology, storytelling, and visual art, aiming to reconcile the lost connection of Southeast Asia's shared heritage.

The main narrative of Jitr surrounds three main domains. Including resounding the suppressed event in the past, raising awareness of unlawful and violent actions by the Thai government or its agents, and upholding decolonized sound. That all inspired by Thai activist, writer, and historian, Jitr Poumisak, or Chit Phumisak (จิตร ภูมิศักดิ์), who was seen as a threat and killed by the Thai government in the 1960s.

The project was developing during artists in residence at Babycastles in Summer 2021, post-pandemic.
Special thanks to @nyfacurrent, @NYCulture, @madein_ny, and @babycastles for the support and for making it possible.

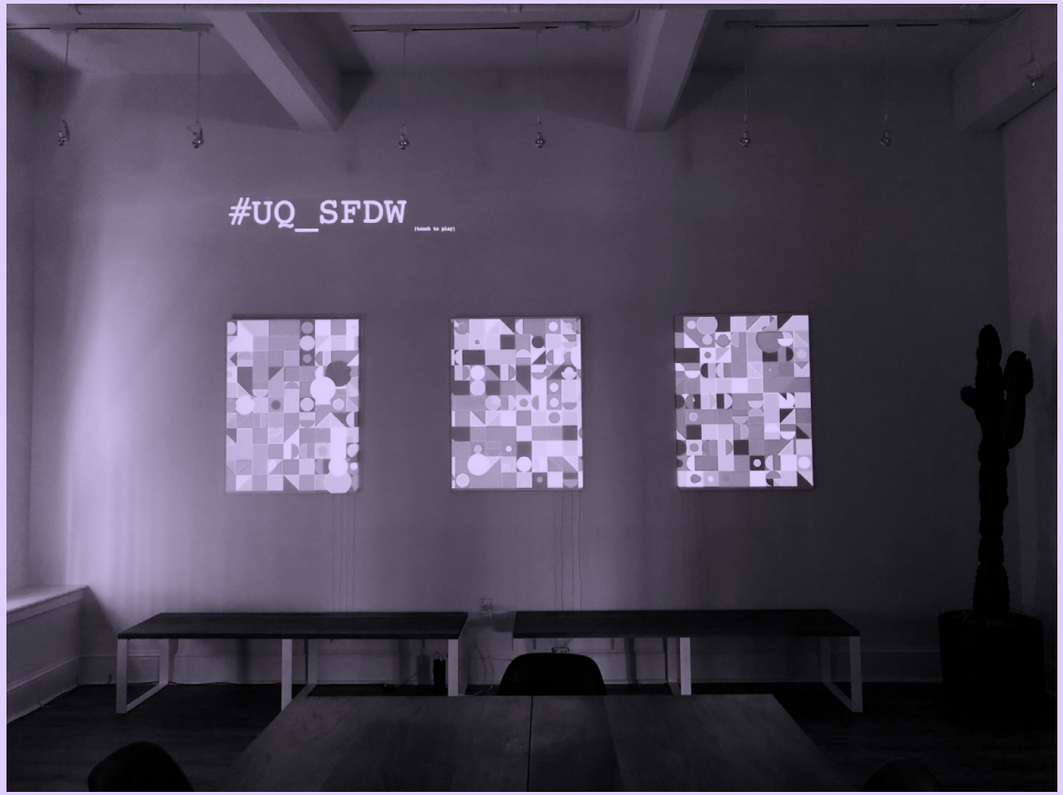
↑
**Jitr performance at Uneven Geographies of Sound at Wonderville photo by Apiwich Bangrapimolpong(@apiwichbang)**

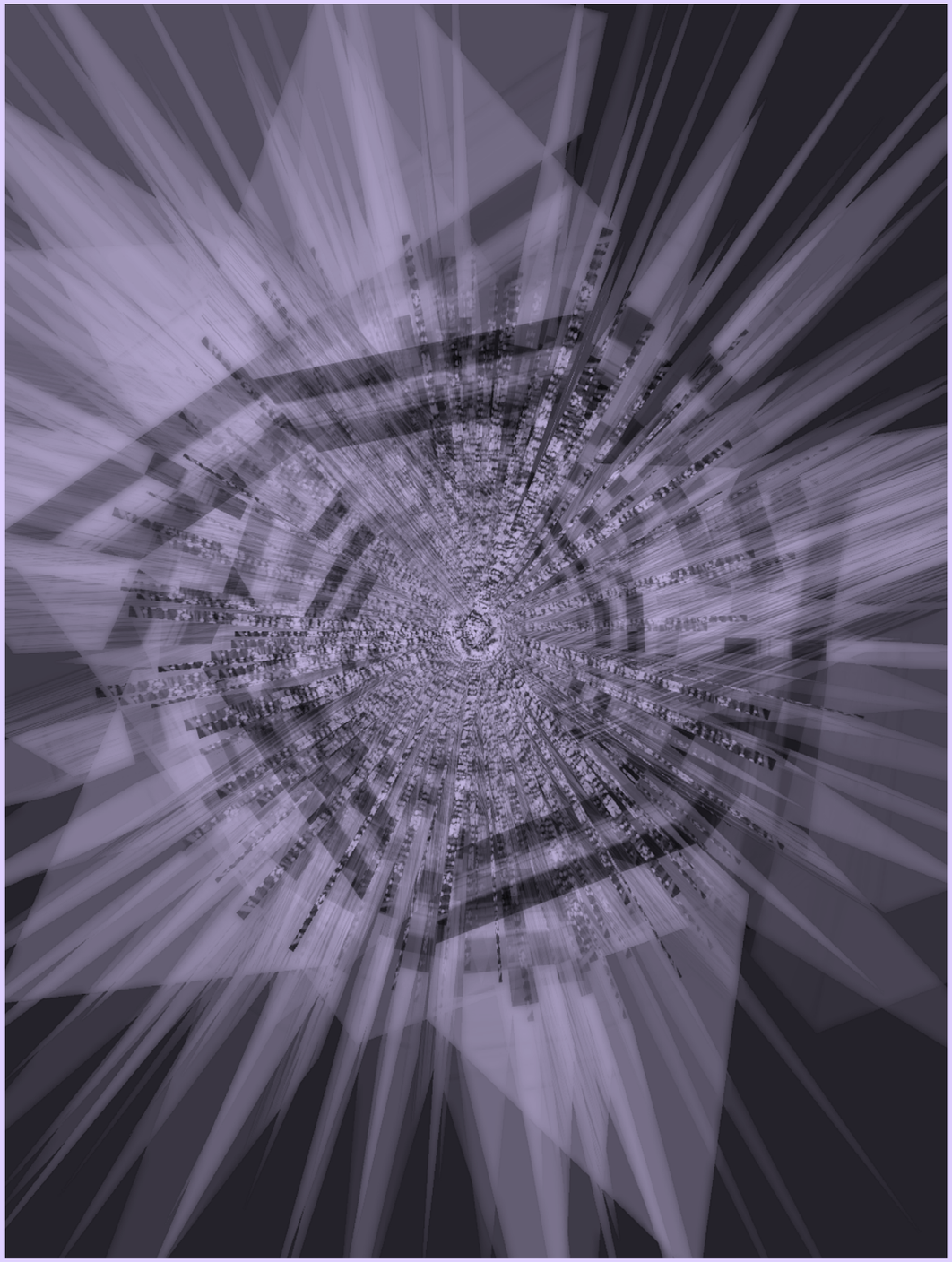
↗
**Jitr performance at Uneven Geographies of Sound at Wonderville photo by Dan Gorelick(@dqgorelick)**

→ **Screen captures from the Jitr performance at Uneven Geographies of Sound event made using P5live and Hydra.**
**– top: Jitr wrote in Thai that can also translate to ‘mind’ in English,**
**– middle: 'a good person' in Thai,**
**– bottom: overlay of video and words in Thai can be read, controlled, and disappear.**


TYPE & FORM
DATA VISUALIZATION & PROCESSING WORKSHOP
KARSTEN SCHMIDT (CEN) 20 21
体用
@toxi x processing x toxiclibs // 2003 - 2011




#UQ_SFDW
(touch to play)






What is the pattern?


# Collective Digital Care

Group Exhibition of USC Media Arts + Practice IML300 & IML400 Class

*Spring 2021*
Instructor: **Qianqian Ye**
Curator: **Qianqian Ye** & **Sarah Ciston**

Artists: **Karen Abe, Ayse Cevikel, Abby Chen, Malia Dollaga, Jennifer Lee, Angela Rong, Katie Liu, Myrah Sarwar, Jonathan Tolentino, Annaliese Tusken, Megan Yeh, Annie Zheng, Qingyu Zhu, Daisy Bell, Jay Borgwardt, Katie Chan, Jacob Pettis, Mahira Raihan, Asha Rao, Emma Sykes**

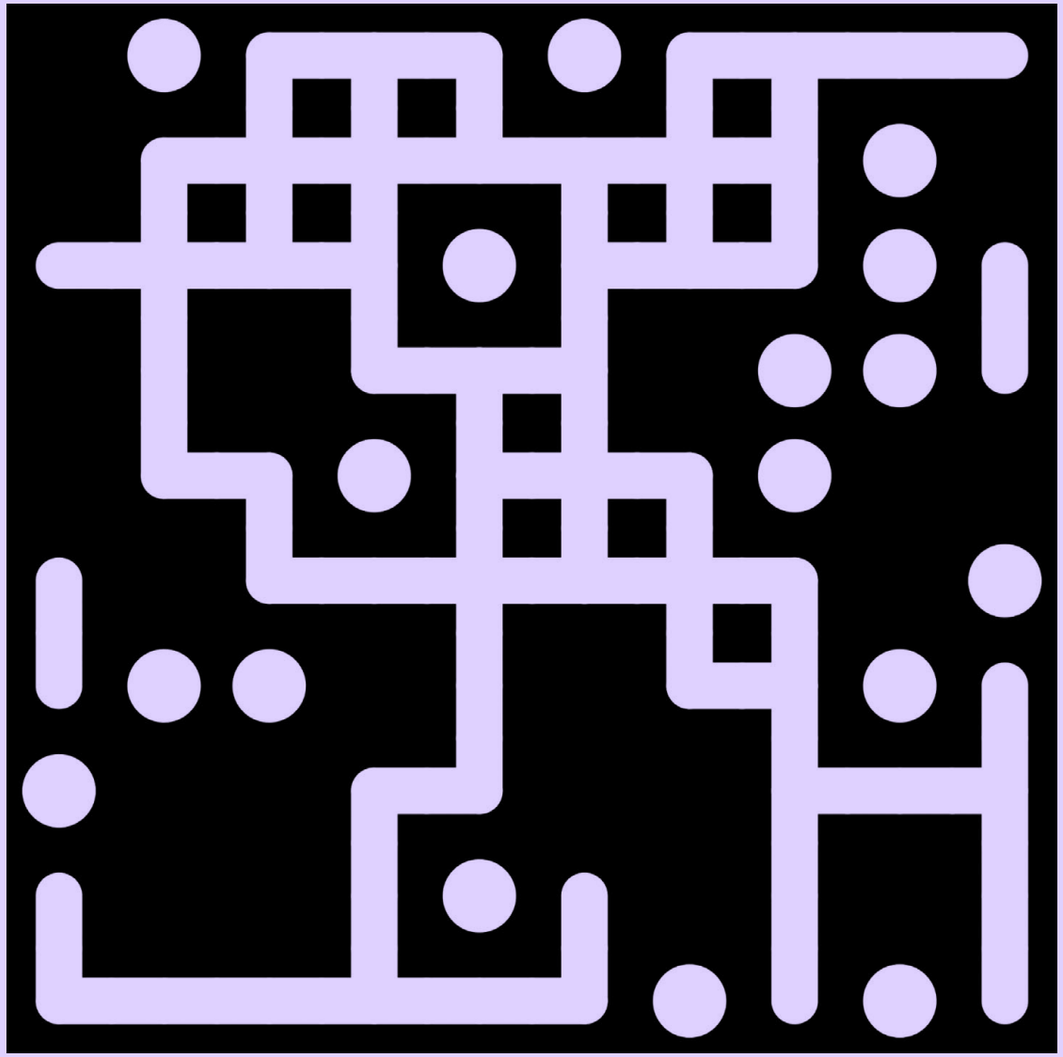
→ newart.city/show/iml300
→ newart.city/show/iml400

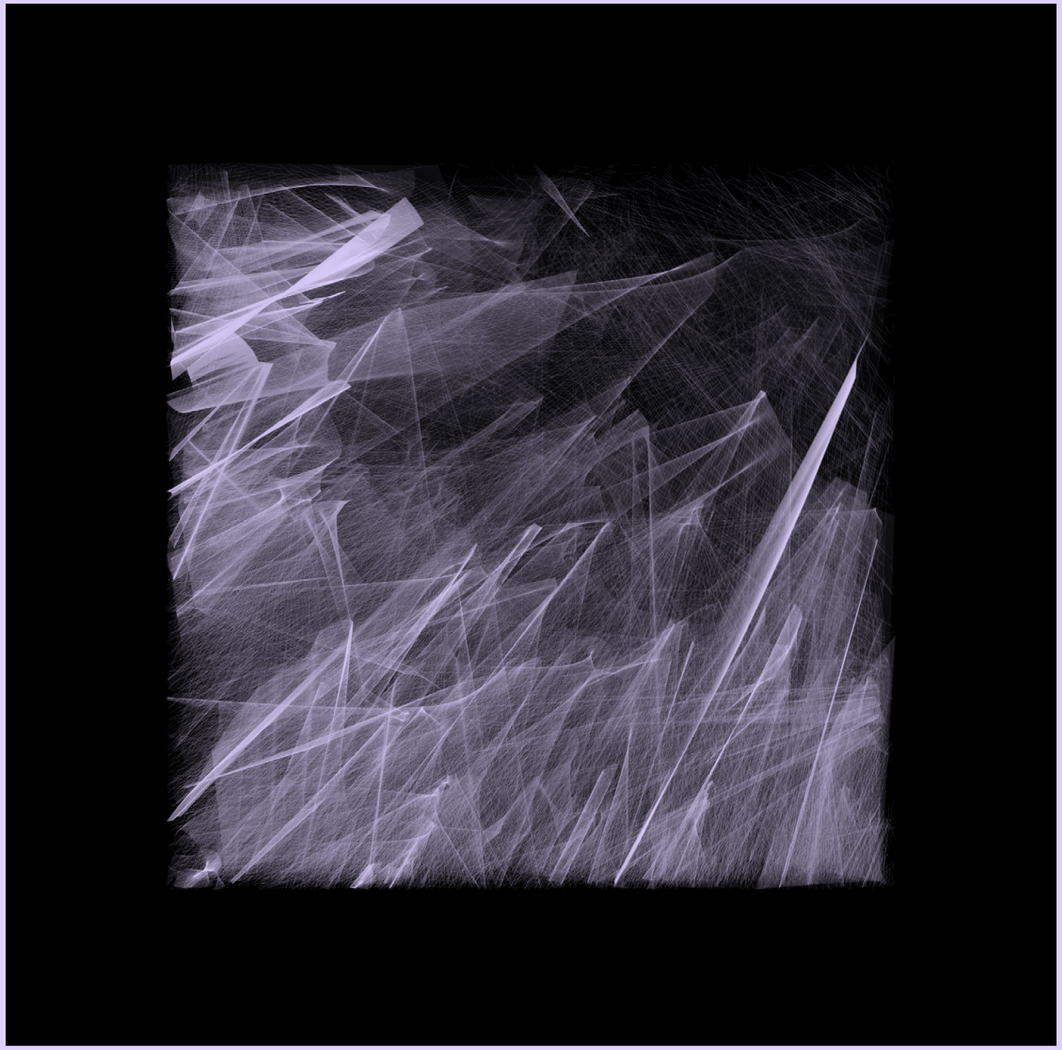
What is collective, what is digital, what is care in reconfigured spaces? How can we use code to express and embrace, revolt and rest?

This exhibition invites visitors to experience the web-driven artworks and artistic research of coders and collaborators in Media Arts + Practice's IML 300 and 400, as we investigated hypernarratives and networked justice (300), cybernetic gardens and other-worldbuilding (400). Along the way, we reconsidered technical approaches to multi-user experiences; interactivity; software libraries; and the aesthetics, protocols, and languages of the web itself.

Collective Digital Care gathers our strength, gathers us together, gathers the digital harvest of code seeds and data packets planted at the start of an uncertain, unprecedented year. It is a record of time spent together and apart, time spent adapting and nurturing, time spent tending and weeding, time spent learning and imagining the kind of interconnected networks we want to hold us together as we emerge, reconfigure, bloom.







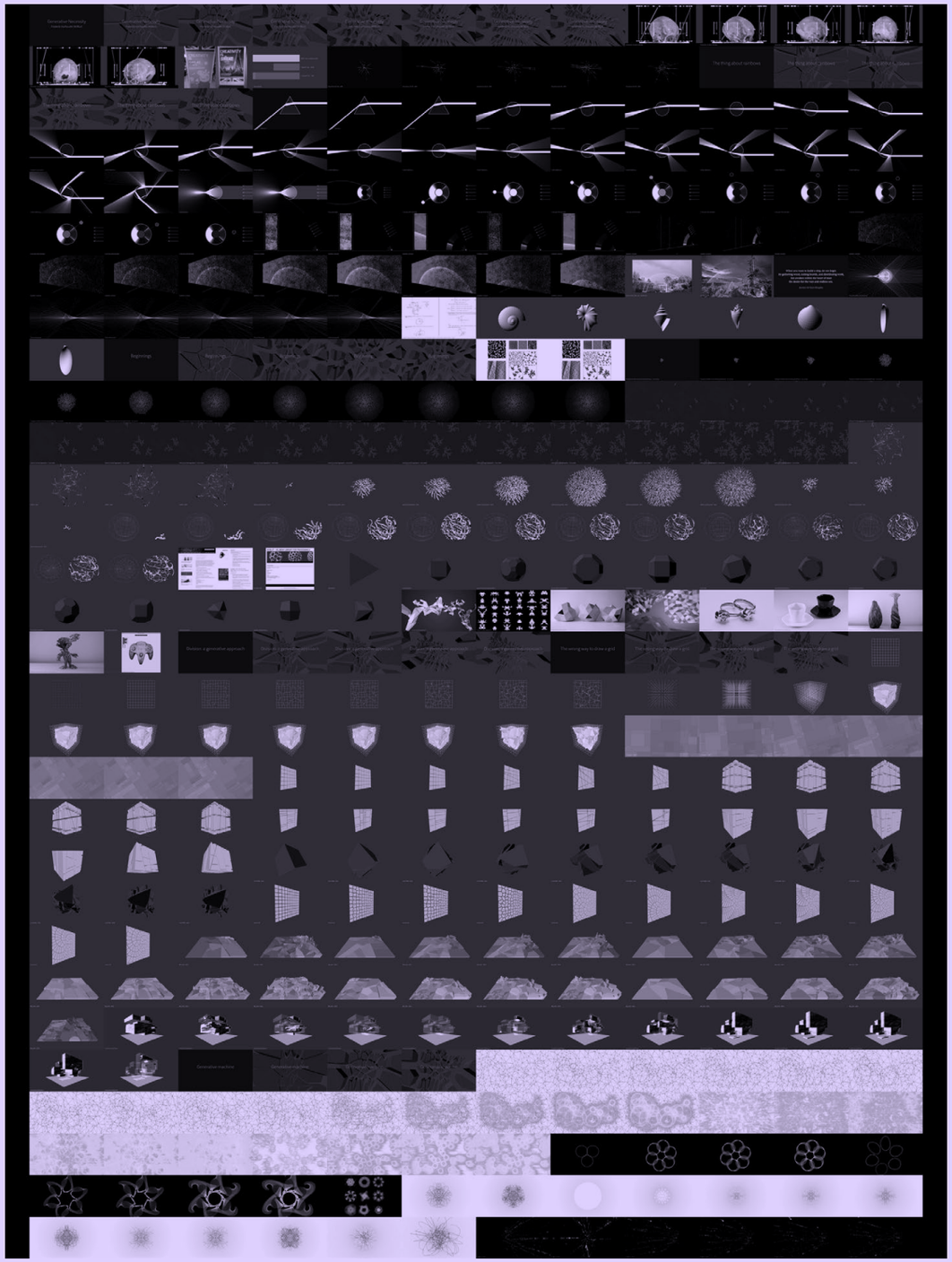
Dear Processing,

You were there for my first animation, VJ set, interactive installation, music video, livestream. I helped you draw lines and shapes, blend colors, read people's code, and work on those little quirks of yours.

Here's to the next 20 years of enabling people to code, creatively.

Jakub Valtar





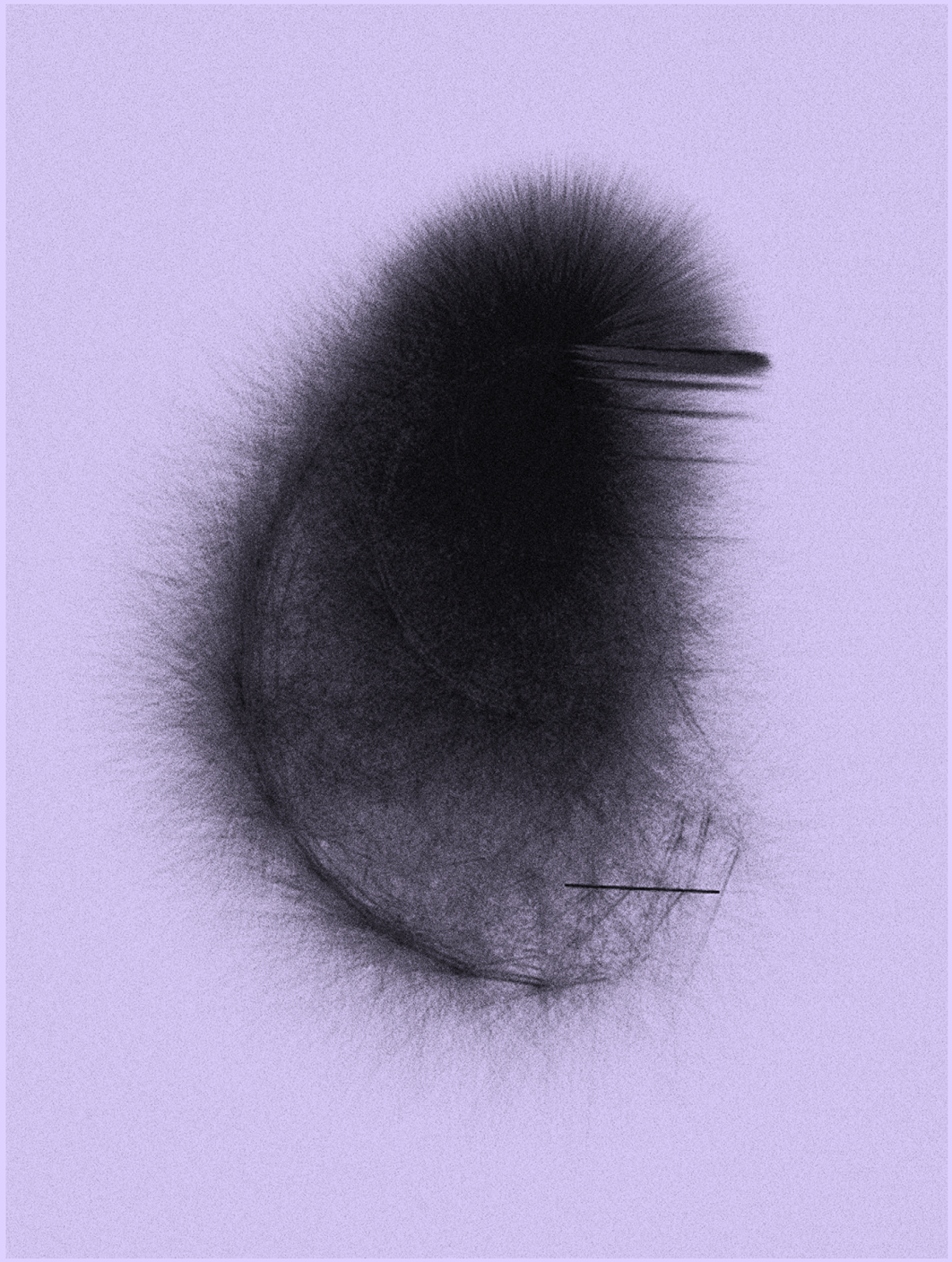


variable ]


For the last two years, I have been using Processing to design the seasonal brochures for Vox Populi, an artist-run space in Philadelphia where I am currently a member. This particular brochure color samples photos of the artists' work, and layers the images with rectangles of the sampled colors. Processing allows me to incorporate nuances and idiosyncracies of these works right into the design, creating a layout that feels much more custom and personal than a more standard design template would for an exhibition.

Image: Gallery booklet for March/April 2020 cycle at Vox Populi, Philadelphia PA. Work depicted (L-R, top-bottom): Kyuri Jeon, *Born, Unborn and Born Again*; Marion Horowitz, *Eat Dirt/Be Free*; Shanina Dionna, *Embryo IX*; Imani Roach, *Old Habits are Harder to Kill than Love*; Kris Ruman & Lauren Fuego, *YYY, How How How*; *The Somatic Stories Project*.



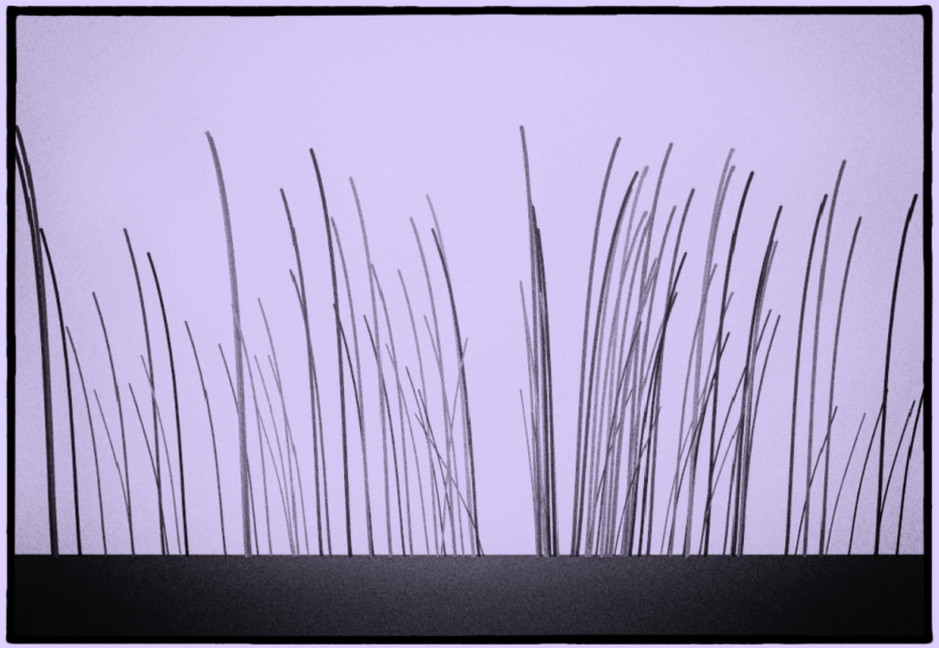


**EXTENTIO - observational drawings**
Sergio Venancio (Brazil)

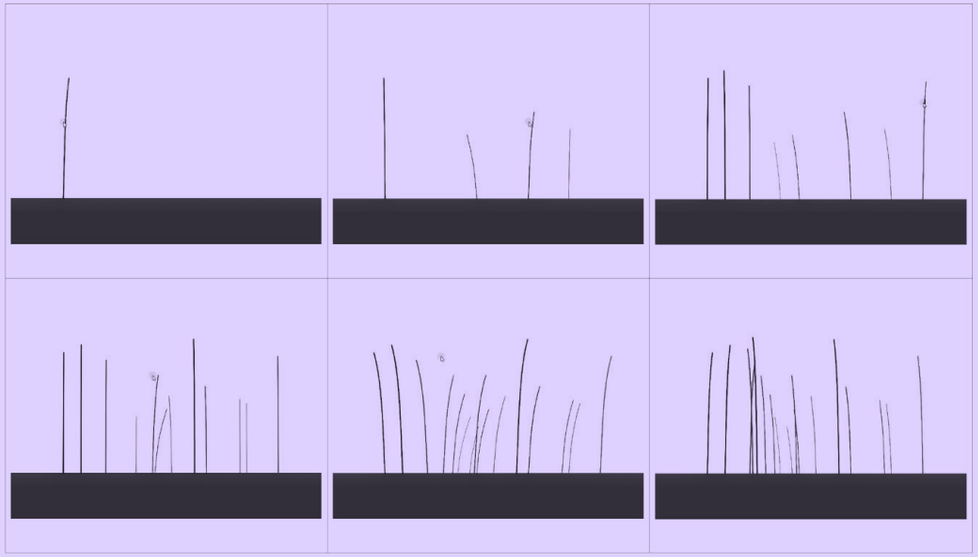
Observational portrait drawings made with straight lines using Processing, OpenCV library (by @atduskgreg) and custom-made AI agents.
Instagram: @sergiovenancio.art



# Waving Grass



Waving grass in P5 with typescript: https://wvinull.wordpress.com/2021/09/11/waving-grass-in-processing-p5-js/



Author: Jeroen Vesseur https://work.wvinull.com/



# Waving Grass in processing (p5.js typescript)

## Table of Contents



## Introduction

I saw the call for contributing a page to the Processing community catalog for the 20th birthday of the Processing Foundation and although my contribution to the community is very limited I was working on a waving grass sketch in Processing and at the same time I was looking for an opportunity to increase my collaboration skills. So this seems a perfect opportunity to give something back!

My contribution has the form of a technical article on how I implemented moving bezier curves in Processing with the intention to resemble 'waving grass' in the wind. The idea started from a completely different angle though. I'm currently working with an AxiDraw Plotter and have been porting some old Processing 3 code to a Java based Processing solution for a generative art project. What I really wanted to achieve was to algorithmically create trees and feed them directly into the plotter. While scouring the internet for information to help me controlling the AxiDraw plotter I stumbled upon Daniel Shiffman's excellent and entertaining coding train tutorials on how to use vectors to mimic real life forces by creating this so called physics engine. This engine was the perfect solution to add to my generative art coding concepts to create trees and use 'natural' forces to alter the shapes of the trees.

### Not trees but grass

Ok that's the why but of course my algorithmic trees were by no means ready yet and the physics engine was a completely new setup for me, so to start I dialed it back a little and tried to create a


A generative catalog displaying a random selection from 10.000 images created by the twitterbot **@n074b07** from April 2017 to August 2018.

All images and the catalog itself procedurally generated with Processing 3.

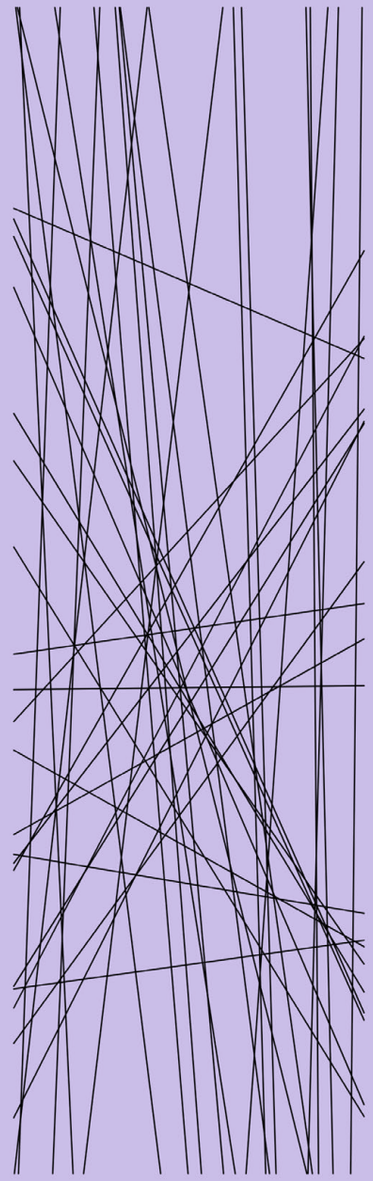
Download the catalog from:
shorturl.at/mszP7

Contact:
José Vicente Araújo
DUNADIGITAL.com
@svcnt

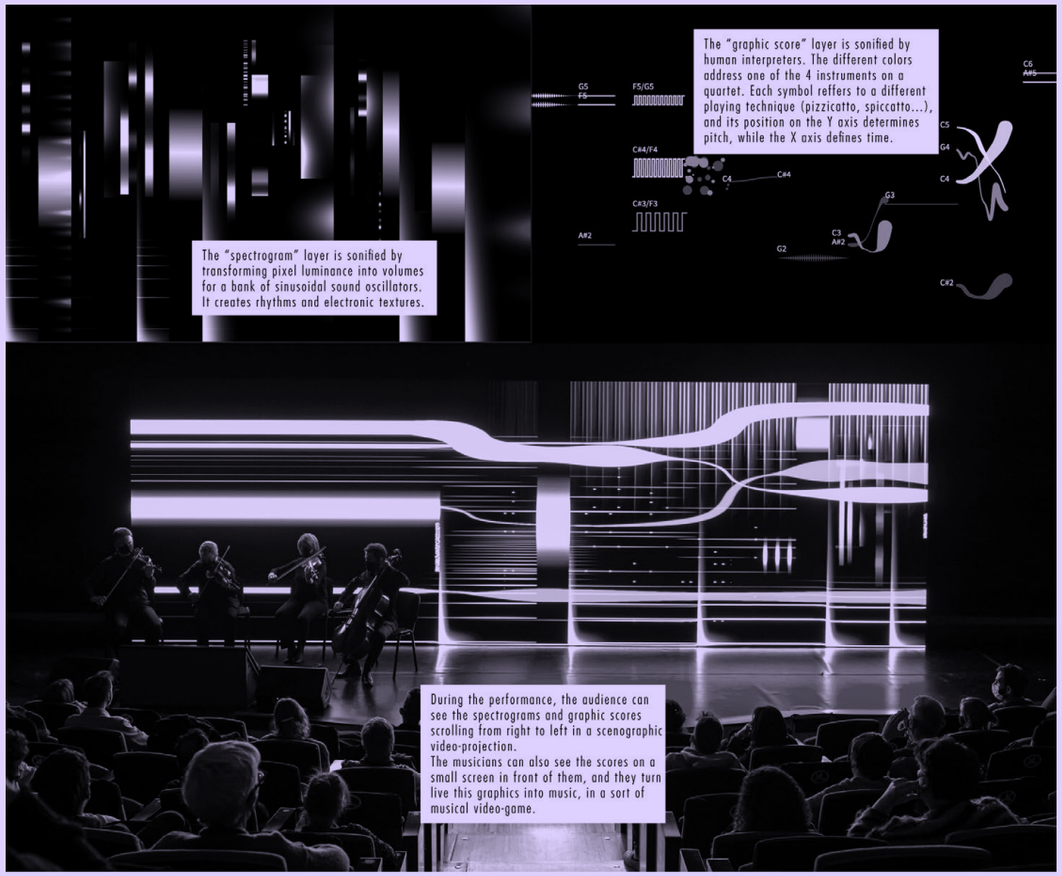


I was first introduced to the idea of creative coding through The Coding Train YouTube channel. After an initial introduction to the world of Processing, I wanted to dip my toes in deeper and luckily, I stumbled across a wonderful 4-day workshop conducted by Ambika Joshi (aka @computationalmama) in 2020 where I learnt the basics of p5.js to recreate works made by South Asian women artists. I'm glad to have taken that opportunity since I now use creative coding quite frequently as a tool to explore ideas.

Giving creative freedom to a computer makes it feel more like a companion helping build visuals. I'm always astonished at the results it can bring about with very little code, and I'm glad to be part of a community that truly celebrates and supports such expression :)



Sketch:
cutt.ly/ironclad_lily
@sairamved




**FORMS–String Quartet** is a live multimedia performance for string quartet, electronic music and panoramic visuals, based on the idea of "image sonification".

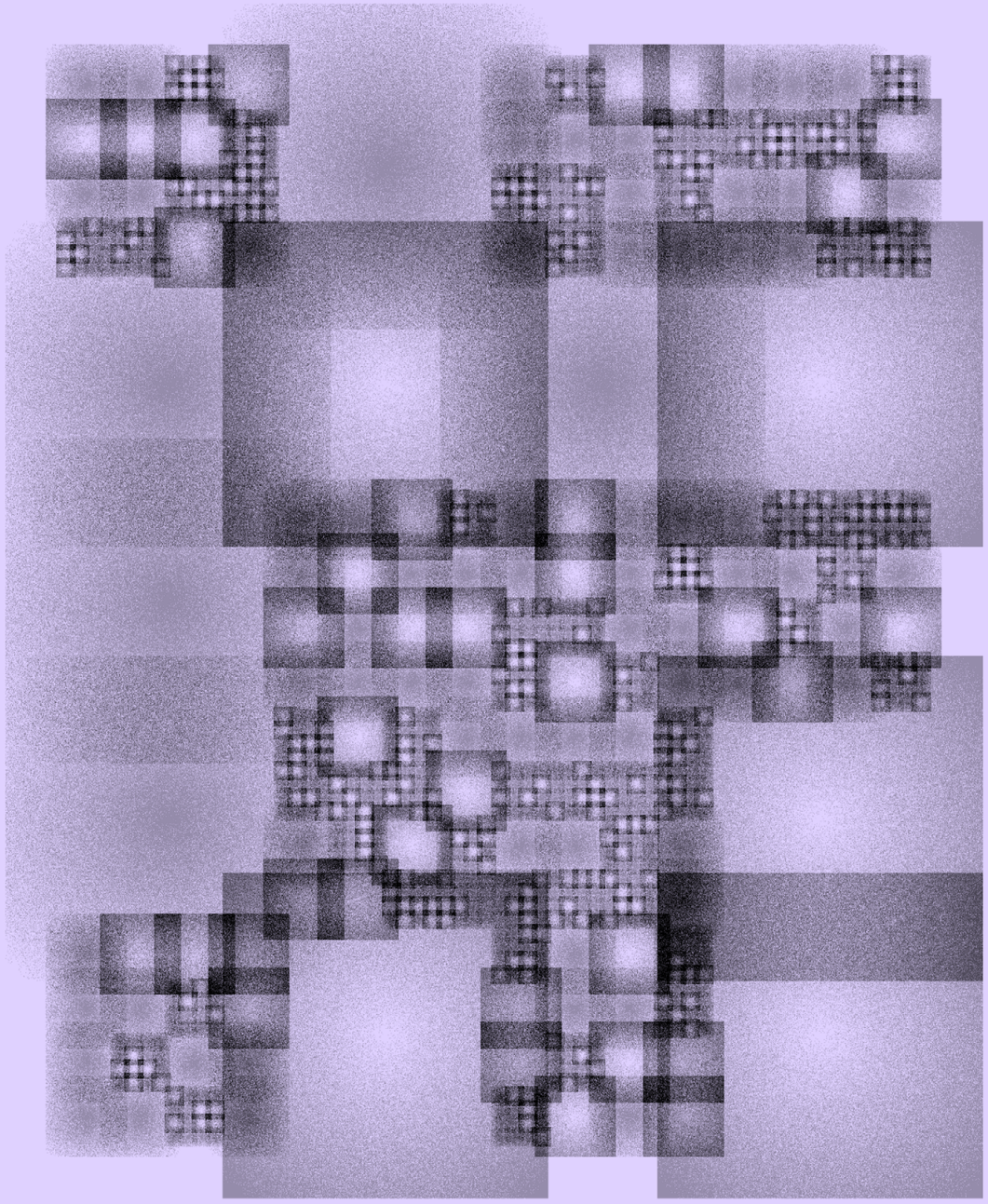
The live musicians interpret a series of graphic scores that in turn build up the visual scenography, offering the audience the power to read the "music to come".

These generative graphic scores, together with the sonified spectrograms that build the electronic music layer of the piece, have been coded using Processing.

The algorithms that generate these scores follow strict musical rules: X axis represents time; Y axis represents tone; each graphic symbol is associated with a different instrumental technique; stroke weights are associated with volume expression... The result is a unique performance that blends music, generative art and technology.

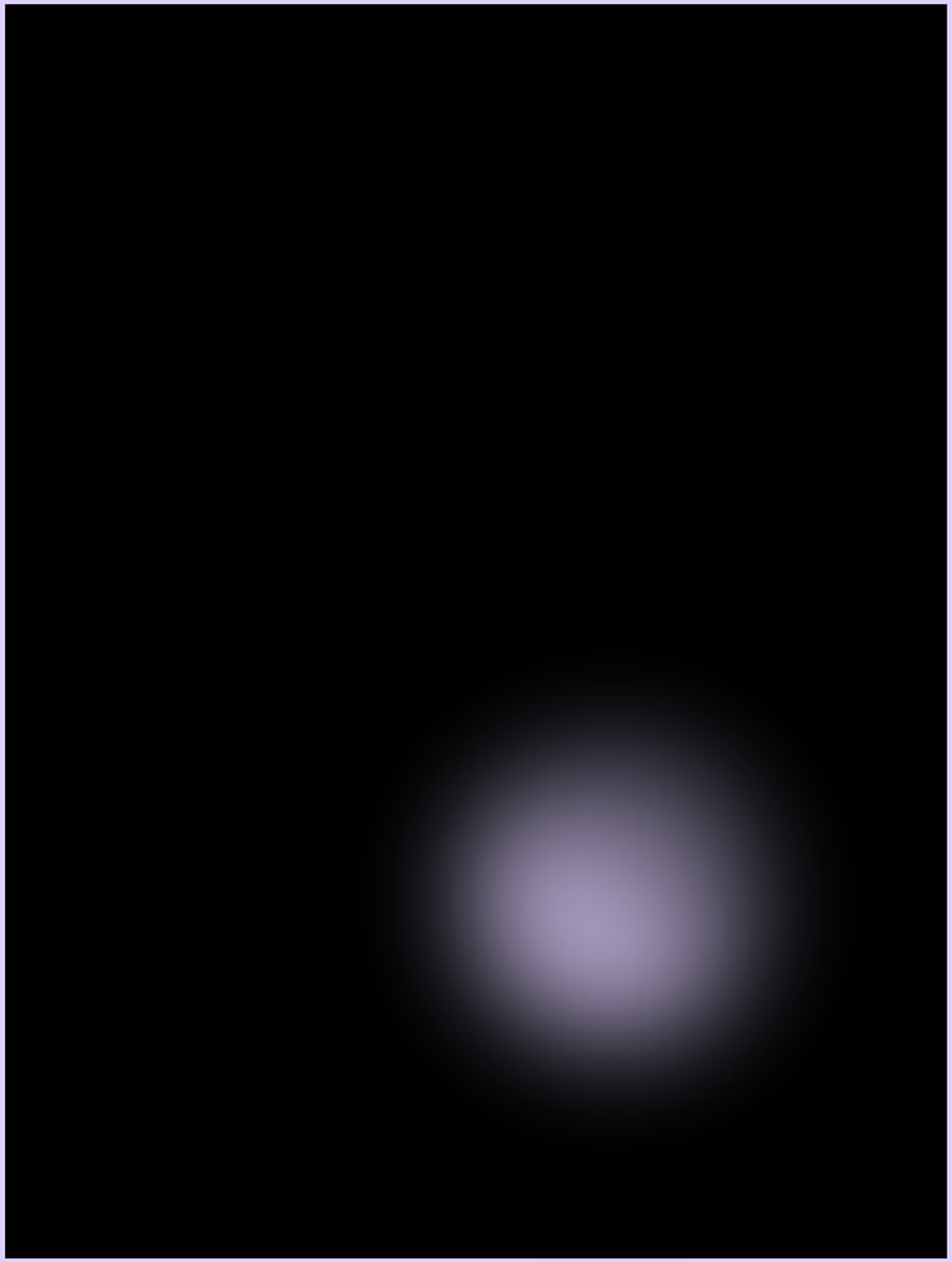
The music on this concert finds a balance between the avant-garde, atonal and experimental approach of the academic 20th century music with the emotional and expressive findings of nowadays urban electronic music.



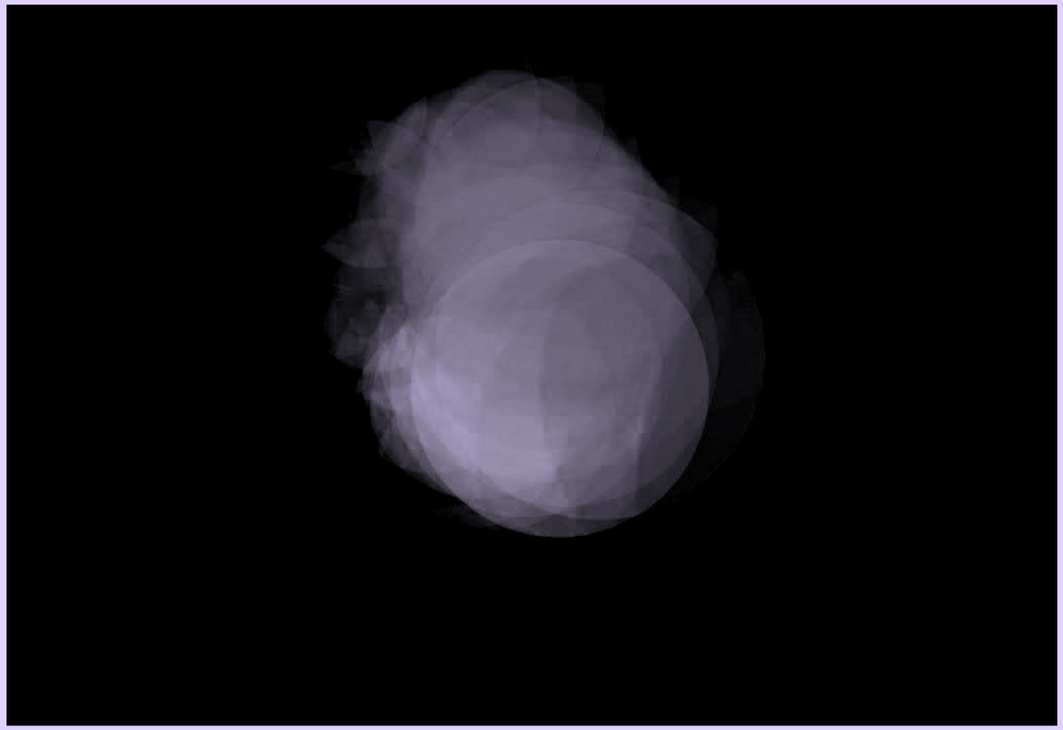


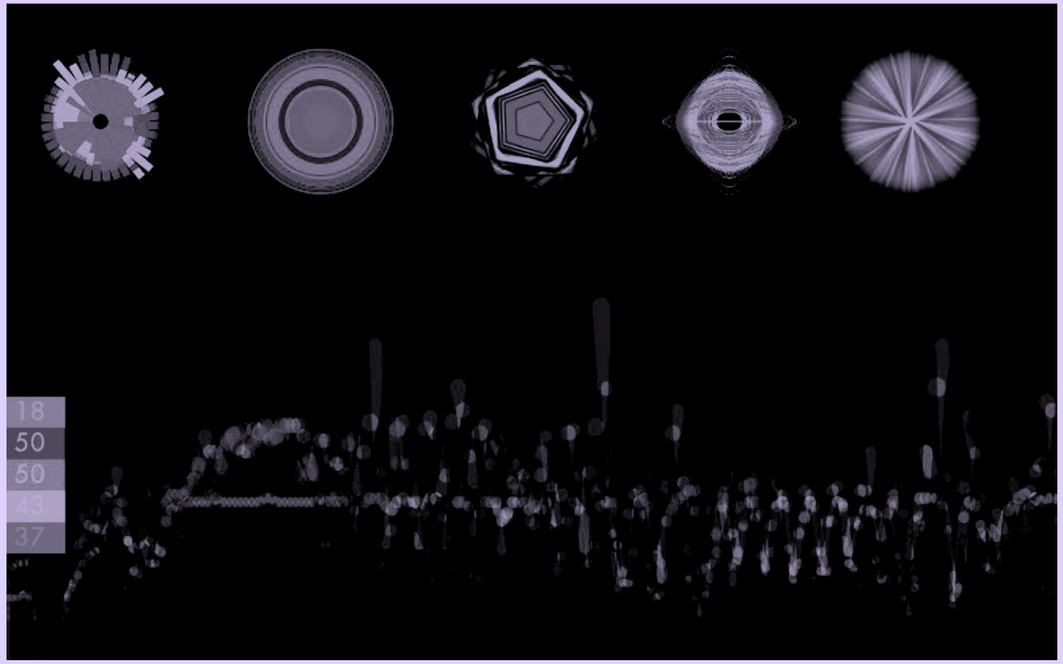
Created with Processing + Python 3, using the py5 library. Code at <abav.lugaralgum.com/selected-work/processing2021>.









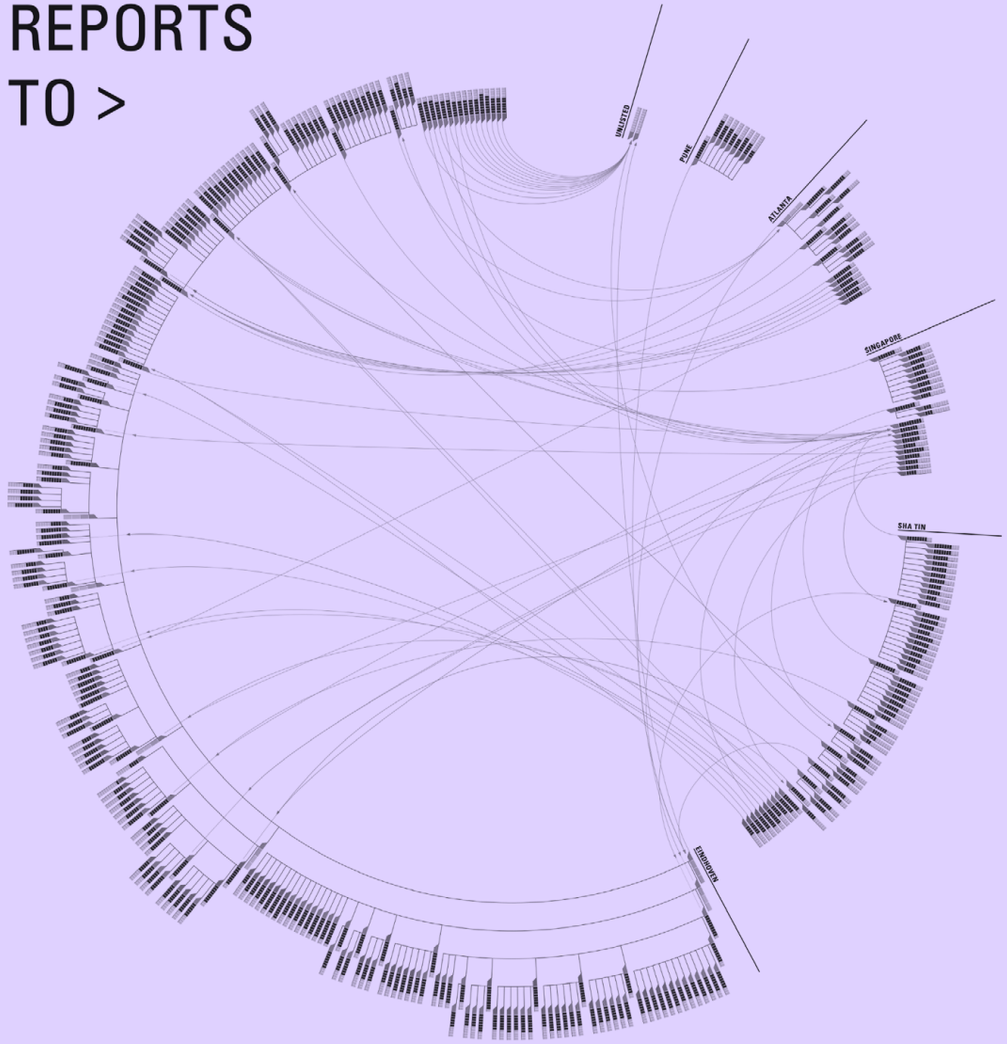

18
50
50
43
37

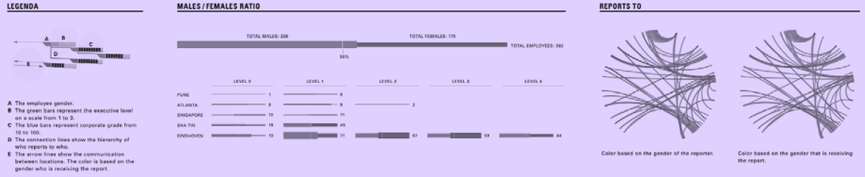
Haoyu Wang,*Starry Everything,* Kinect and Processing,Dimensions Vary

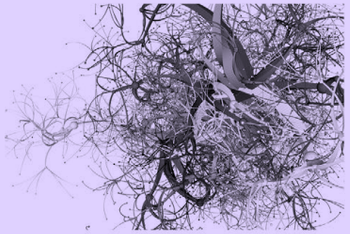
Haoyu Wang,*Starry Everything,* Kinect and Processing,Dimensions Vary

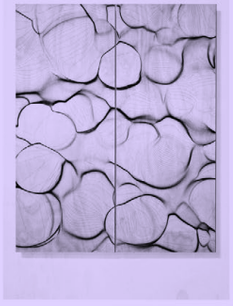


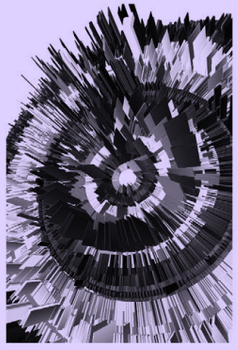
# REPORTS TO >



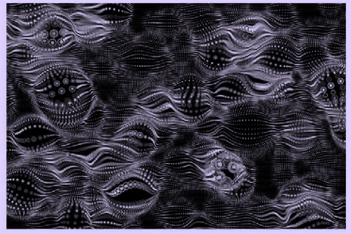







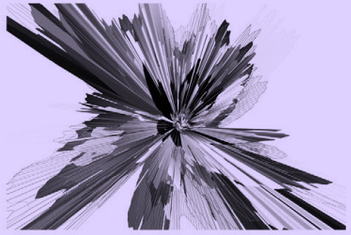


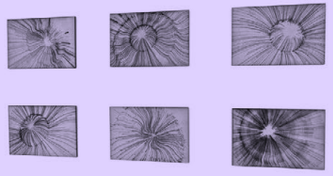
# MARIUS WATZ

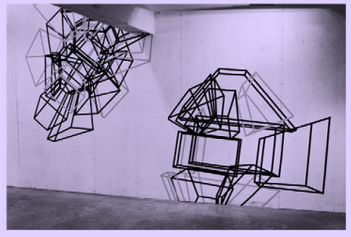


Processing user since 2003. Teacher, artist, designer.

Generative systems, realtime animation, data visualization, prints, 3D printing, laser cutting, light installations, plotter drawing, wall drawings, media facades.









## A SENSE OF COMMUNITY

It's quite astonishing to think that twenty years ago, Processing was but an idea germinating in the minds of two young university students, Casey Reas and Ben Fry. What appeared as perhaps just another tool for creation at that time, has evolved into something quite extraordinary and I'm sure far more empowering than they could have ever imagined. Their vision for making code a more accessible medium has become a shared vision for an increasingly active and international group of educators, artists and designers. Many of these people embark on a multitude of activities that both support as well as help develop the project and on many levels because Processing goes beyond the idea of just another production tool. It is infused and driven by an open source working ethos that upholds and encourages values worth sharing, cherishing and indeed celebrating.

To partake in this particular celebration, I have decided to share with you a little story. One that made me not only realise the importance of Processing but also gave me the strength to become, in part the person I am today. This story is also a part of the Processing story and I like to think it played a small yet important role in the community's development. It all began back in Paris in 2009 whilst working with the organisers of the Offff festival for whom I'd been given the mission to do a little side event. At the time, I was working as a journalist, specialising in motion design. A lot of my work involved interviewing artists and designers, a wonderfully fulfilling job and an experience that gave me unique moments of insight into some of the best creative minds out there. Three important figures were to cross my path and influence me in my decision for what I wanted to do for Offff.

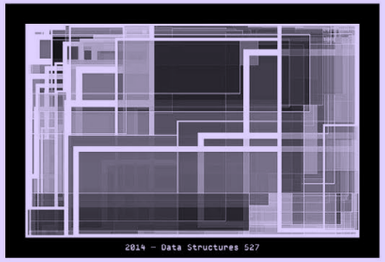
John Maeda, Joshua Davis and Karsten Schmidt are three key figures in the Processing story but at the time I knew very little about them and even less about code as a creative medium. Quite incredibly, I had the immense honour of meeting these people as part of my job and through those wonderful encounters, I was introduced to a completely new world of creation. One that immediately fascinated me. What perhaps excited most was that I saw the immense potential for artists and designers to express themselves through a medium that was imbued with ideas that transcend the simple concept of a tool for creation. I saw a way of thinking, a philosophy of life if you like, a community driven effort for the common good of all those willing to learn but also share in their creative endeavours. This was an important moment and it was to shape a lot of my thinking for the rest of my professional life. The first chapter opened with an official invite for Karsten Schmidt to lead a masterclass workshop in Paris as part of Offff.

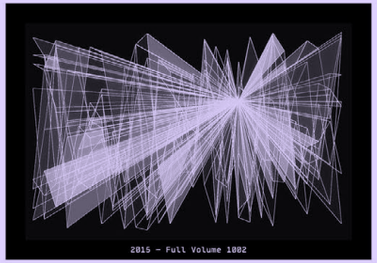
At the same time I was on the scout for some home grown talent to accompany Karsten as well as help me understand what the scene was like in France. It didn't take long before sitting down for a beer in northern Paris with Julien Gachadoat, to convince him of the initiative. Two beers on and we'd set the date for a beginners workshop. Some months later in 2010, the first edition of what became to be known as Processing Paris took place in a suburban quarter of western Paris. It brought together roughly forty people from more than six different countries and from a variety of artistic disciplines. They were hungry for knowledge and also excited by this rather humble event.

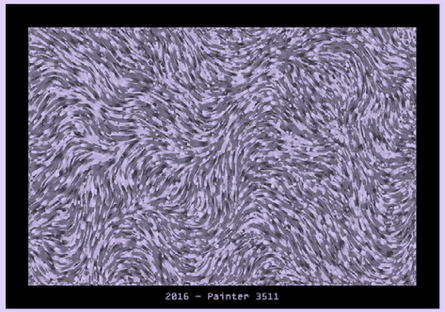
Indeed that excitement was more than palpable, it became tangible with a flood of positive email feedback and a resounding yes for a second edition. I took up the challenge, organising a third workshop to take place as well as extending the duration of each. Quickly, a small community of international people began to support this humble endeavour. I set up a non-profit organisation and began to organise as well as lead monthly meet-ups. There was no mistake, something was happening in France and beyond. Processing Paris became Processing Cities; Berlin joined, Ghent joined, Frankfurt, Austin and Chicago popped up amongst many others and even Julien invested his spare time in setting up a series of events in his home town of Bordeaux.

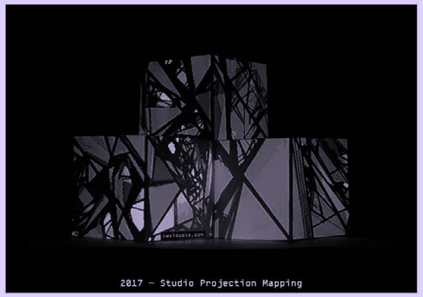
I succeeded in organising five editions of Processing Paris, finally deciding to let go after I left the capital, taking on other responsibilities in 2015. It was an honour to realise this little project even if it was small scale. Funnily enough, it was perhaps that aspect that many appreciated. We were a human scale event, almost family size really and yet it had important ramifications for those who came. What made the event special was that it brought a community together and it was a catalyst in the building of relationships that would extend beyond, some on a personal level who have become good friends, some on a professional one making great art together. I am extremely grateful to all those wonderful people who supported Processing Paris, who came year after year, who helped out, taught, shared, played the music and laughed into the early Spring evenings we spent on the canal Saint Martin after a hard days coding. And it is this that I remember and cherish most of those days. A sense of community, of belonging and the simple beauty of bringing people together, but also watching them depart together, occasionally hand in hand.

We need this vital community spirit more than ever in these times. This may sound a rather ominous or misplaced statement for some, perhaps even a little too grand for the small scale community that Processing is in relative terms. Yet, when I see, read and hear about how that community has developed over the years, I'm reminded of its importance, not just on a personal level and not just as a tool, but indeed as something that extends far beyond with values that I can only commend and adhere to. May it continue to be.



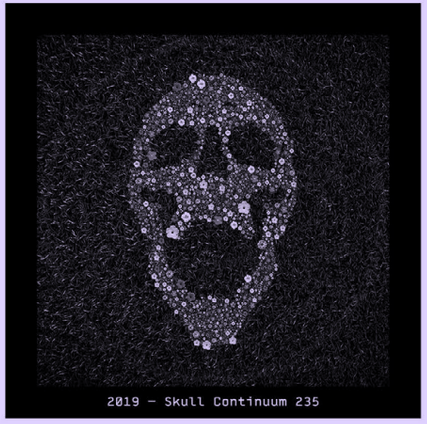



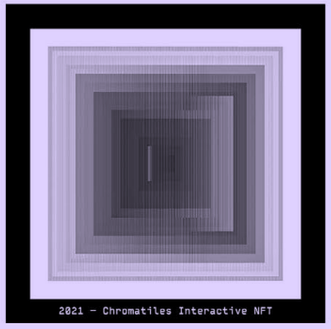




www.cwvisuals.com

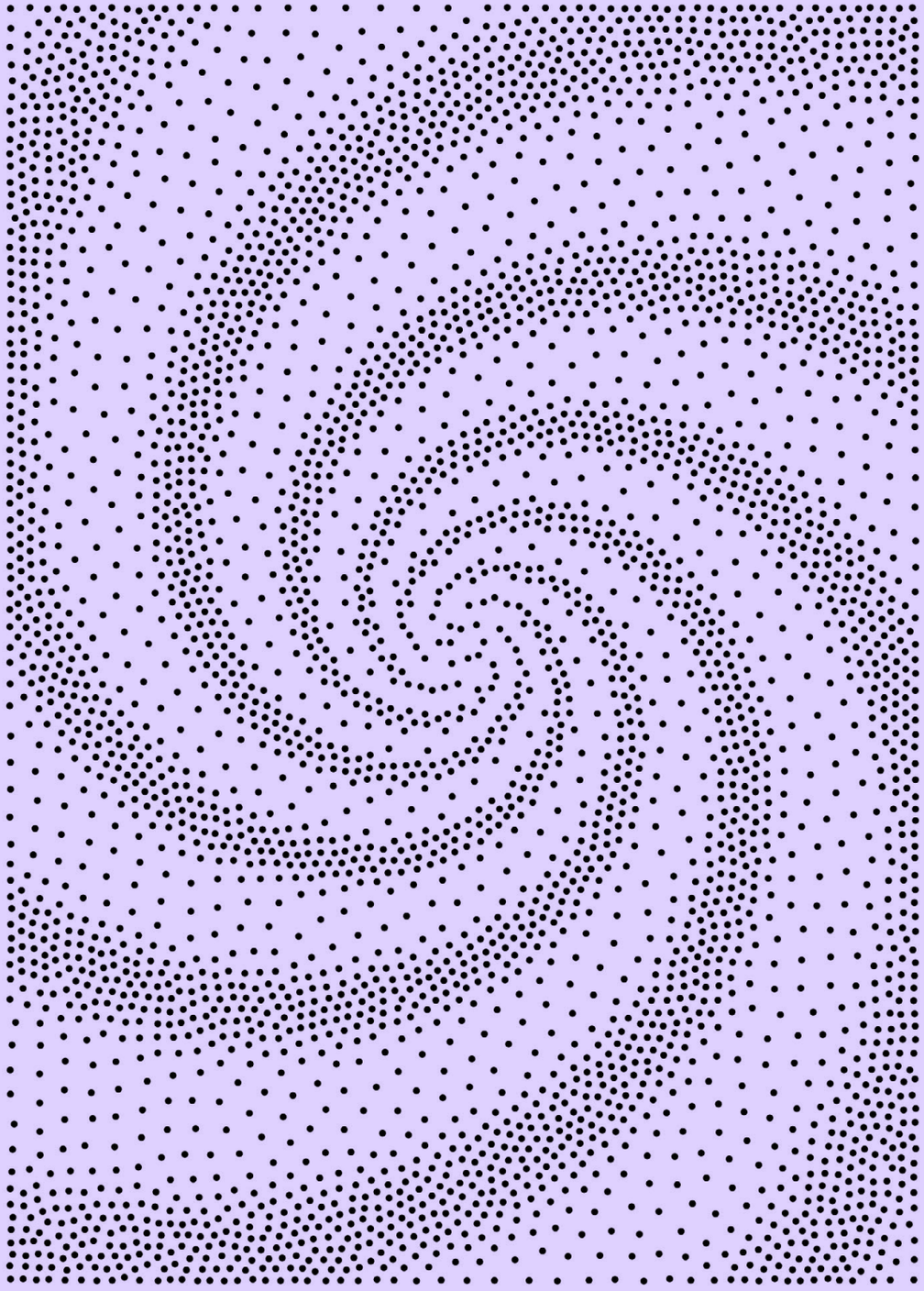
Chris Welch

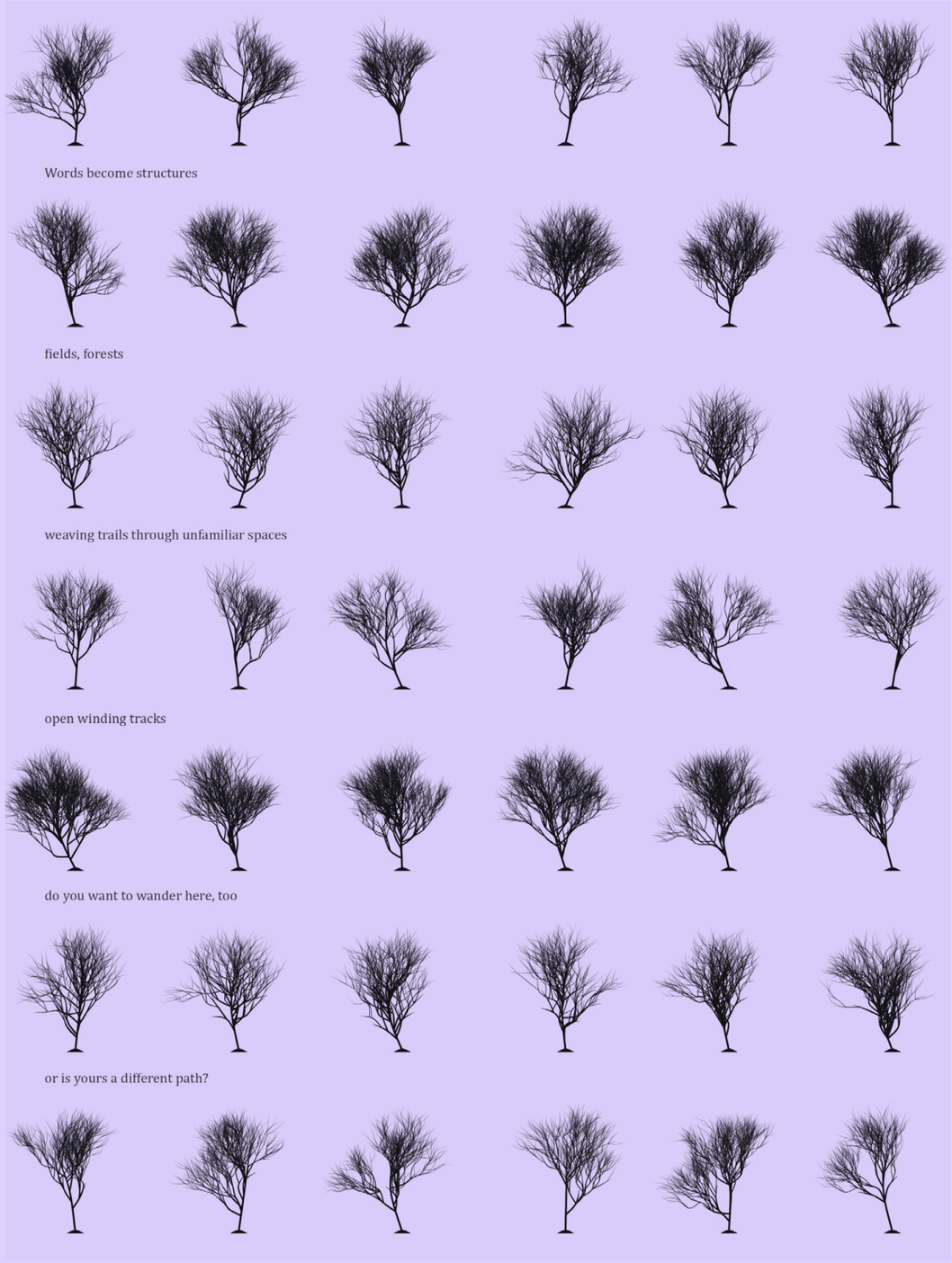














```
void setup() {

  /*
   * When I met Casey in Ivrea, Italy,
   * I had just finished grad school,
   * where I had tried creative coding
   * but in unstructured ways.
   *
   * I've forgotten how I was introduced
   * to proce55ing but I remembered
   * the room where I stayed up all night
   * and really learned to code.
   * I sat next to a blue stove
   * that looked like a perfect cube,
   * inside a modernist dormitory
   * that looked like a perfect arc.
   *
   * So I learned proce55ing
   * in such an unforgettable circumstance.
   */

   size(20,1);
}


void draw() {

  /*
   * In the next few years,
   * I was fortunate to be involved in projects,
   * considering that I could barely write
   * a recursive function without crashing the computer.
   *
   * I made an applet of a pond of fish,
   * one of the early sketches
   * in the Exhibtion section of the website.
   * I led a small team to complete
   * the first Chinese translation of the website.
   * I collaborated with Casey, Jared, and Robert
   * on the Software Structures project
   * for Whitney Museum Artport.
   *
   * There were so many brilliant artists and hackers
   * in the proce55ing forum,
   * which I visited daily
   * and learned so much from the commnuity:
   * vector field, braitenberg vehicles,
   * nonlinearity, fractals and more.
   *
   * (I could almost write a recursive function
   * without crashing now)
   *
   * With regrets, I became less involved gradually.
   * I guess I've "grown up", got a job, and so on.
   *
   * But thanks to Processing and its community,
   * the joys of coding stayed with me.
   * It's like learning poetry or art history.
   * When you understand them, they enrich your life
   * in deep meaningful ways, and urge you always
   * to see and consider and create.
   *
   *
   * For these and many more reasons,
   * thank you Processing!
   */

  background( 255 );
  fill(20,20,20);
  rect( random(20), 0, 1, 1 );

}
```

@williamngan, 2021/9/30 03:10AM

Decentralization was (and still is) the hope of many of us hacktavists, early builders and punks of the first generation of online weirdos, the first children to grow up with the internet and the last generation to remember life before it existed. In those early days, we saw the internet as a real promise, that it would make information free, democratize media, grant new forms of economic self-sufficiency. Many of us believed we could change the way the world works from behind our glowing screens. Some of us actually did that, and many things have changed, but not exactly in the ways we imagined.

**What does Processing mean to me?**

Being a contributor to open source projects we know that all of these tools are built by people. We have built systems to address ethics, we know if a system is lacking a moral code it is our responsibility to code it.  I know this because I am a member of the Processing community along with other open source projects that helped me to build code with ethics at the forefront of our communities. When we build rules and logic into systems, we imagine a better future and listen to one another around fears, joy and possibilities for the things they want to see in the world.

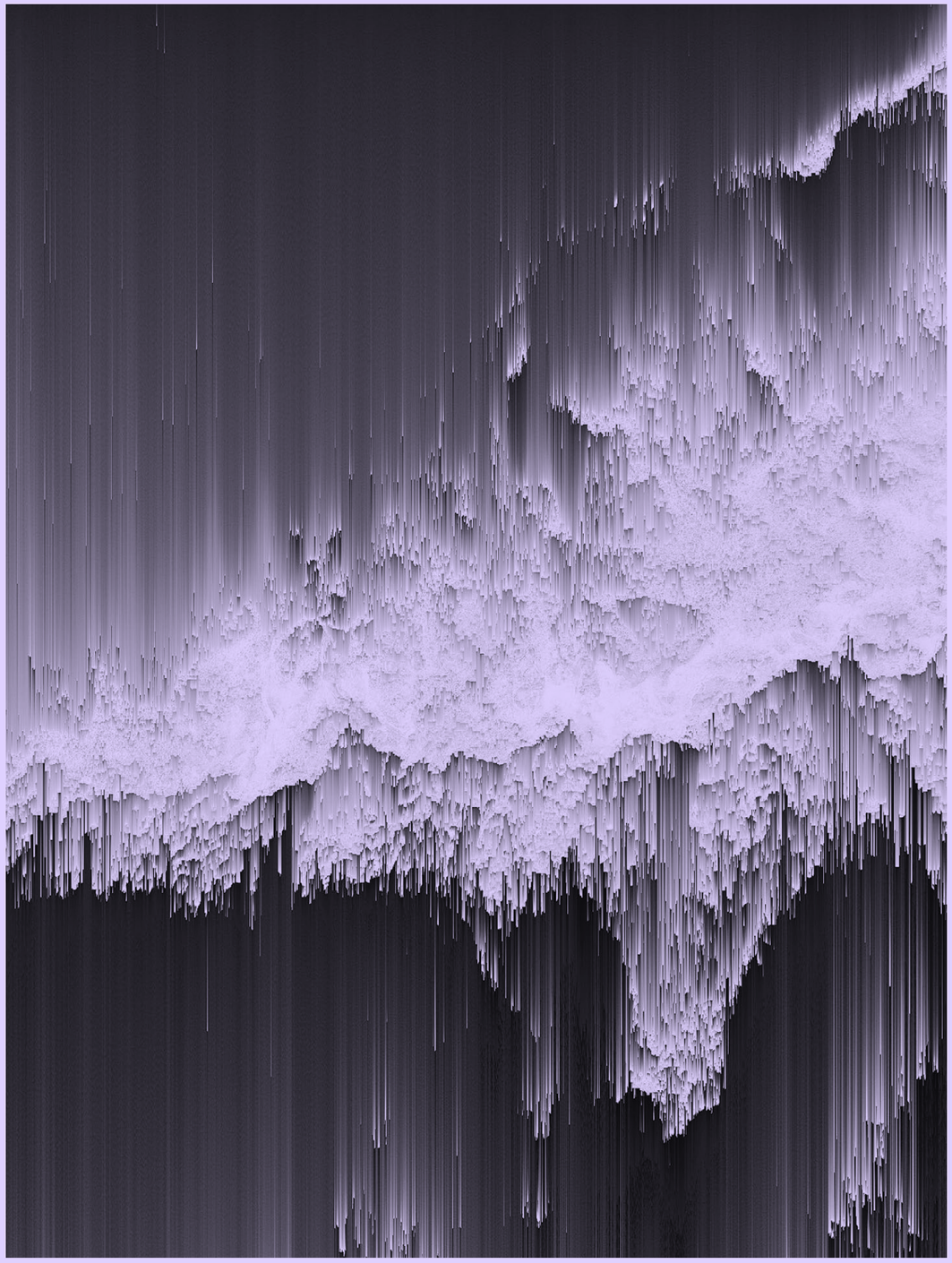
I created Wampum.codes to address exactly this issue.
Wampum.codes is an ethical framework for software development based on the principles of co-creation as understood by my people (Seneca-Cayuga Nation of Oklahoma, Indigenous.)

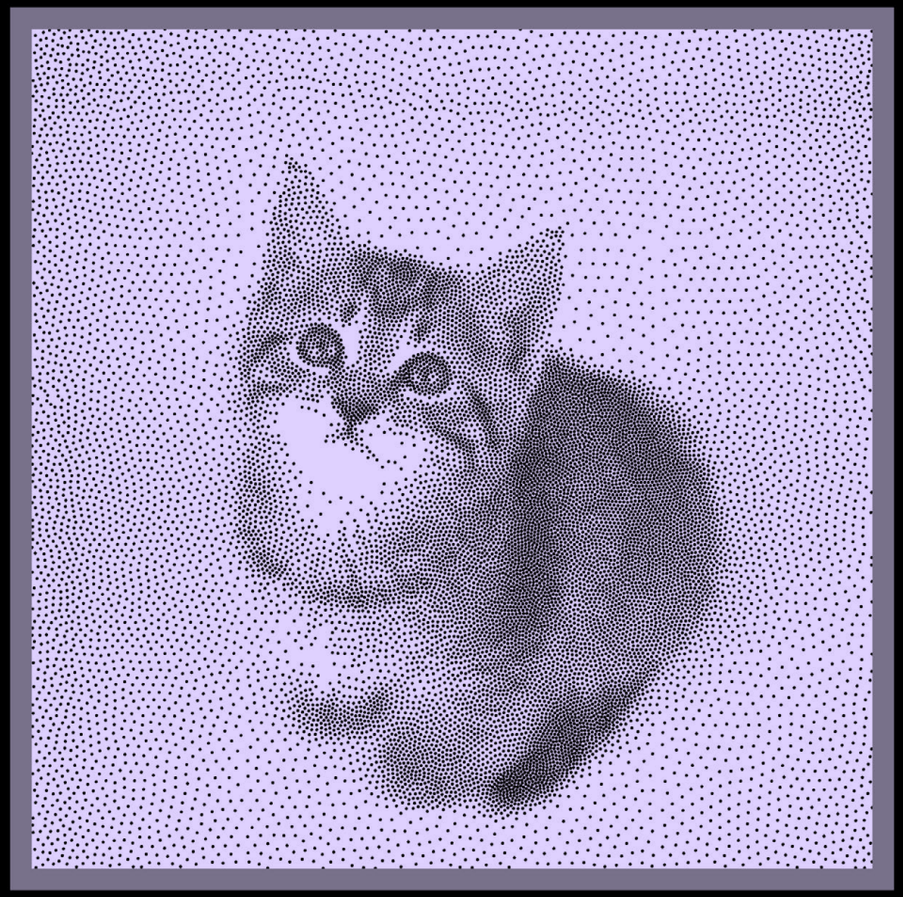
Like all members of the Iroquois Confederacy, we made wampum. A lot of people have a misconception about what wampum is - they think it was a form of currency. It was not currency - we used it as a tool for recording and regulating the different political and economic agreements that governed daily life. It was a decentralized means of recording contracts, something like a pre-Colombian blockchain, that encoded not just financial transactions, but also ethical values.

The project of Wampum.codes is to try and imagine how we can weave ethics back into 21st century technologies. We can embed these values as dependencies in code the same way we do in the rest of our package.json

We don’t have to wait for experts and policy makers to make these decisions for us. As good as their intentions are, they don’t know what to do, and they will always be a step behind. This is our field, we know how to do it, and we are the ones who need to step up. By implementing a decentralized protocol around ethics in software, we can step in the right direction. **We live our lives according to a moral code. The time has come to code our morals.**





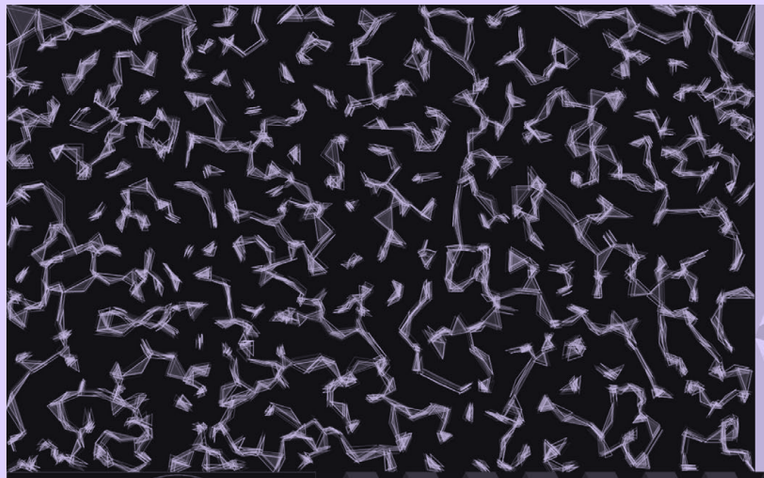


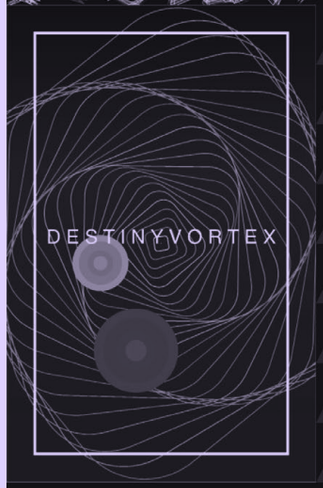
COMPUTATIONAL IMAGE STIPPLING

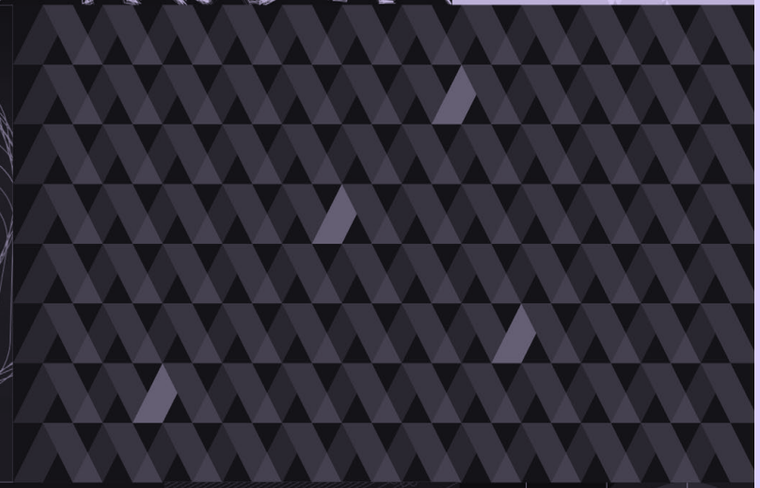
16,384 POINTS, COMPUTED USING PROCESSING

@ARTIXELS



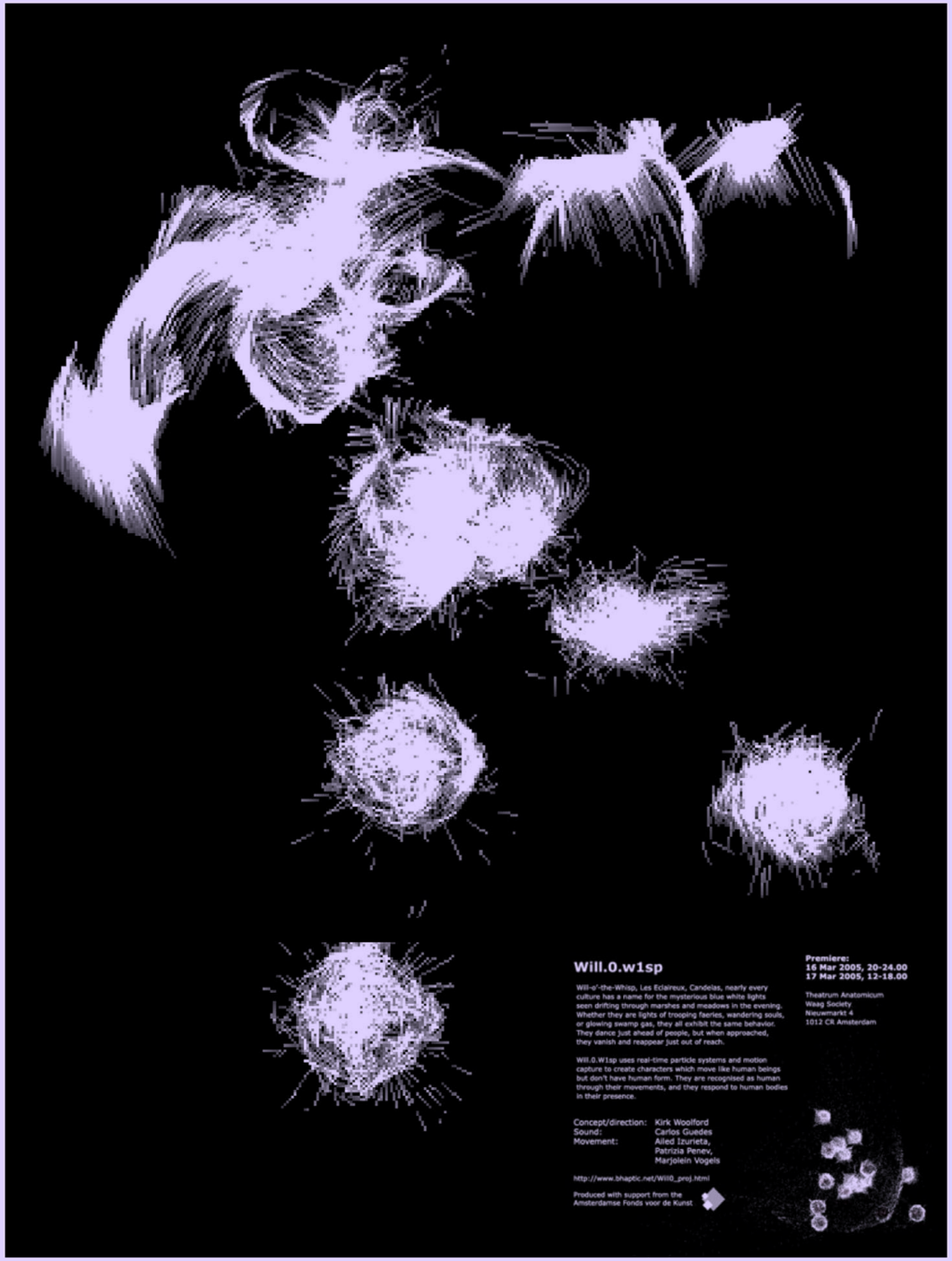




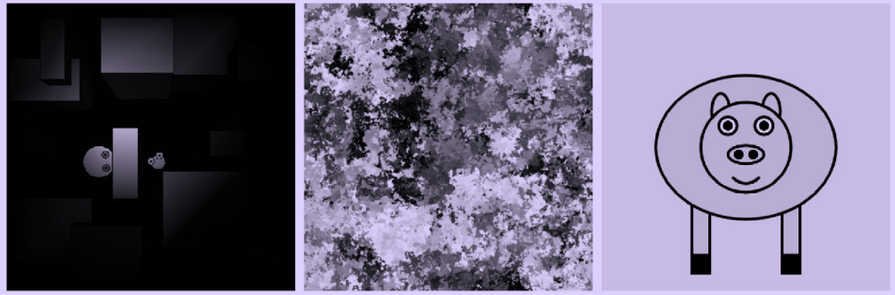
**Queens College, CUNY**
**Creative Coding Course**
**Professor Danne Woo**
**creativecode.dannewoo.com**

Displayed here are student projects created using Processing and p5.js in the Creative Coding course at Queens College, CUNY. These pieces are created by Adolfas Lapsys, Kiana Wong, Kristine Albaran, Michelle Burk, Monique Lee and Zeran Li.

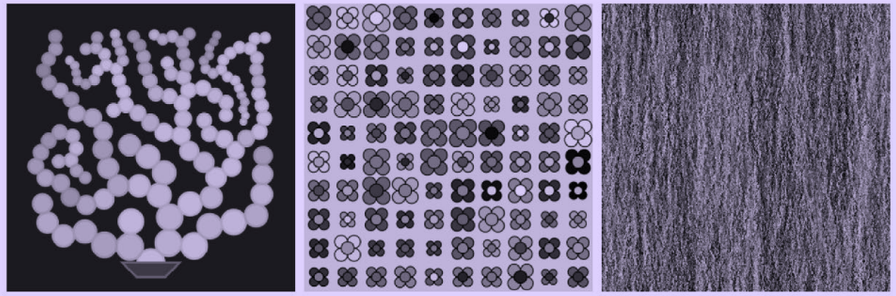




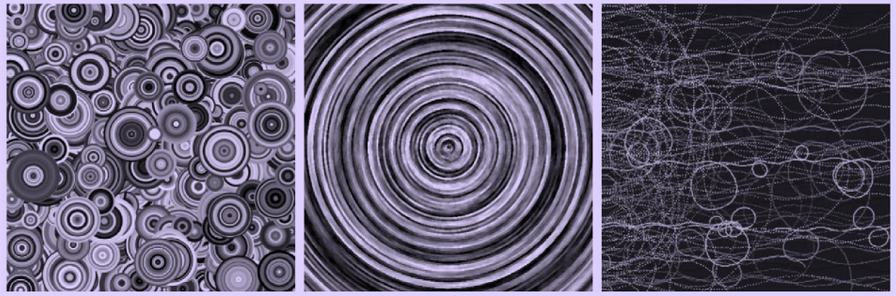

Whenever I introduce somebody to Processing and p5.js, I start by explaining that they're more than "just" coding languages or libraries. Sure, it starts with the code, but more importantly, Processing and p5.js are a **community** of people: learning, struggling, creating, and expressing themselves in new ways every day.



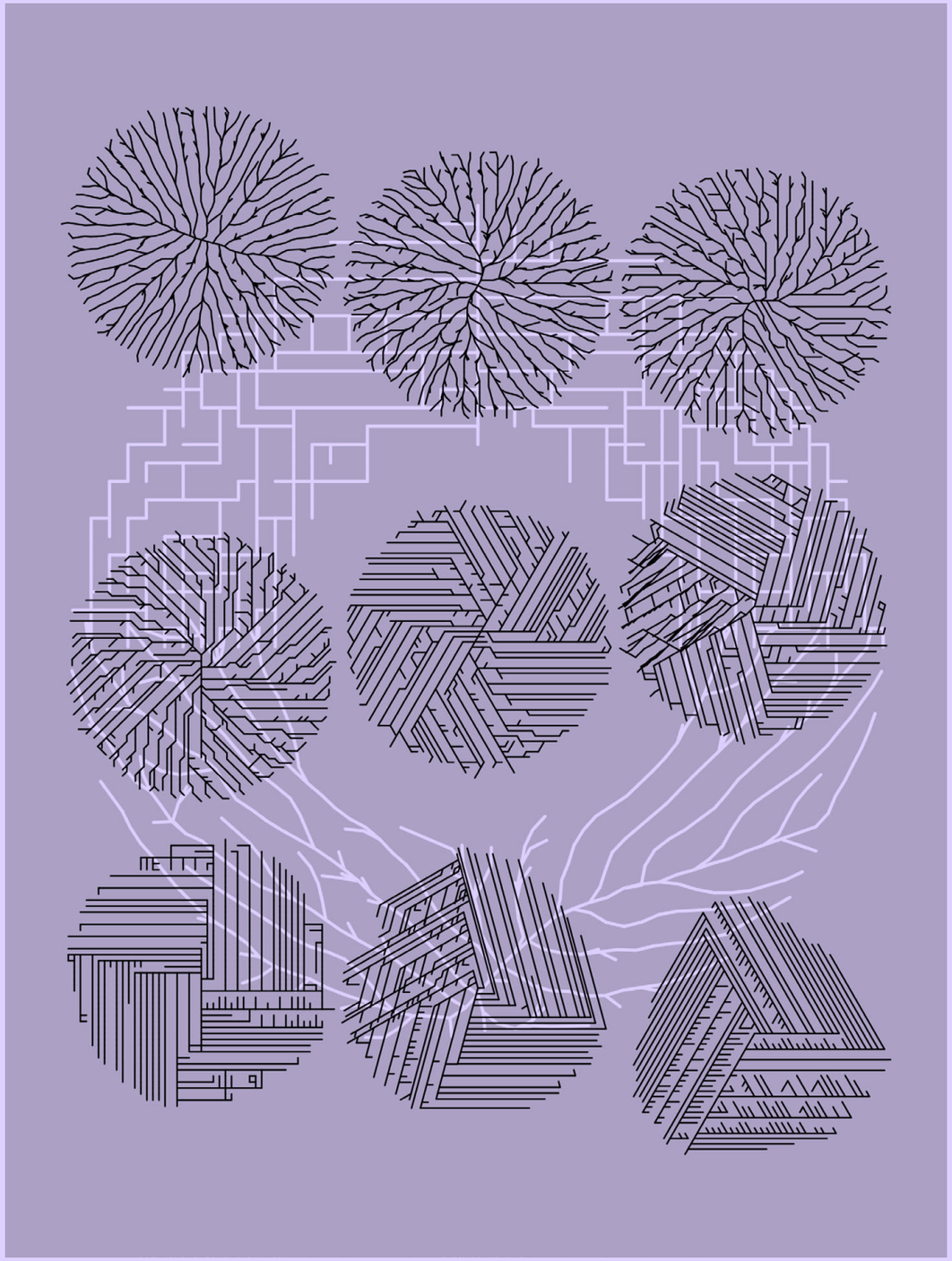
I found my way into this community through the Java game dev world. I learned Processing for Ludum Dare 21 and instantly fell in love. I started spending time on various forums and Stack Overflow, and eventually I branched out from games, into creative coding, outreach, and education. I've even met people from the Processing community in person and at events like CC Fest.



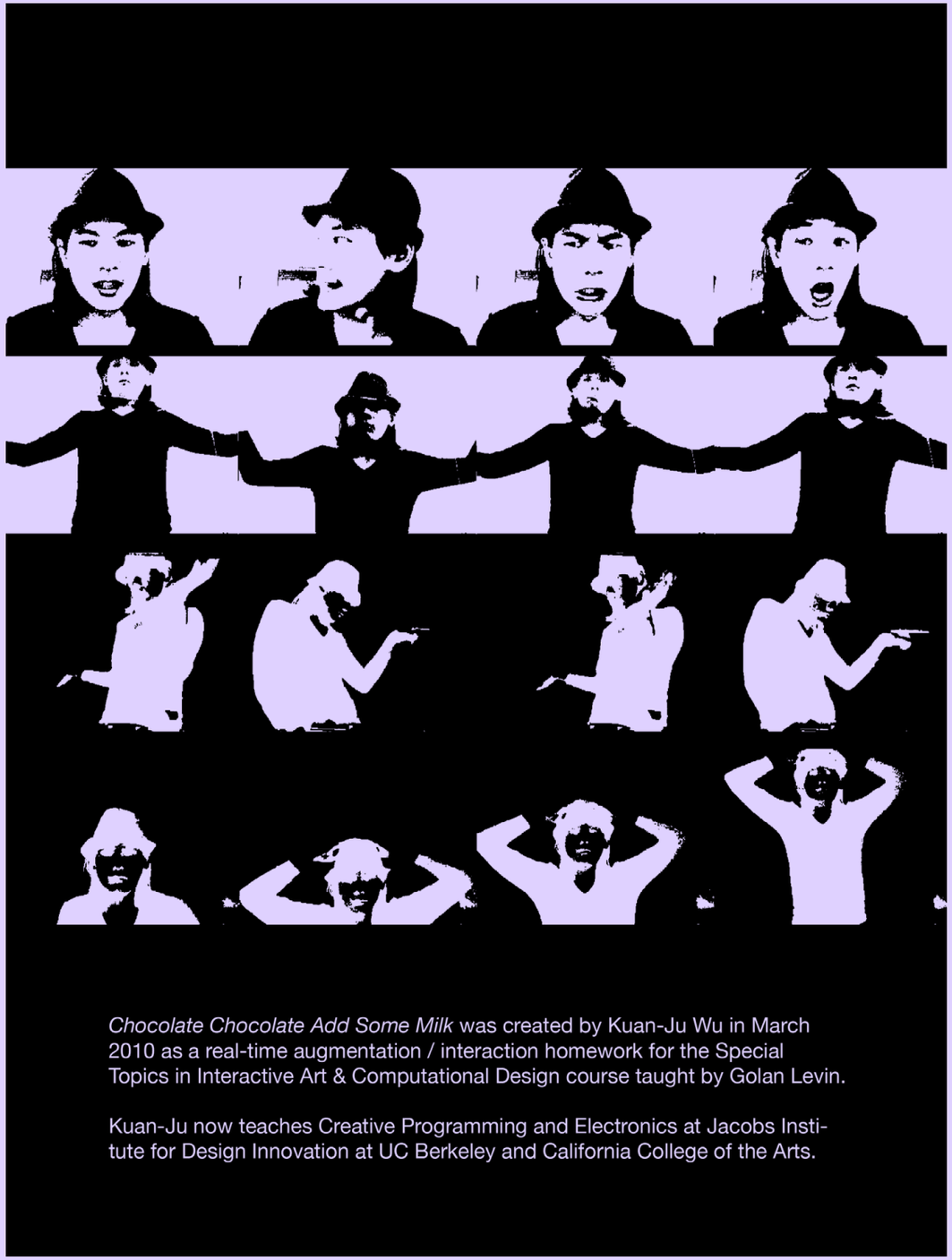
Processing has let me direct my creative energy, has helped me figure out what I want to do with my life, and has helped me succeed in my career. But more importantly, Processing has given me a community of folks who understand that code is more than just ones and zeroes. It's people.



Thank you to Casey, Ben, Lauren, Cassie, and Dan for building this thing we love so much. And special shout out to George, Saber, Kelly, the Upperline Code folks, and everybody hanging out on Twitter, Stack Overflow, and the forum. You're what makes Processing a community for me. Thank you. <3 Kevin Workman - HappyCoding.io

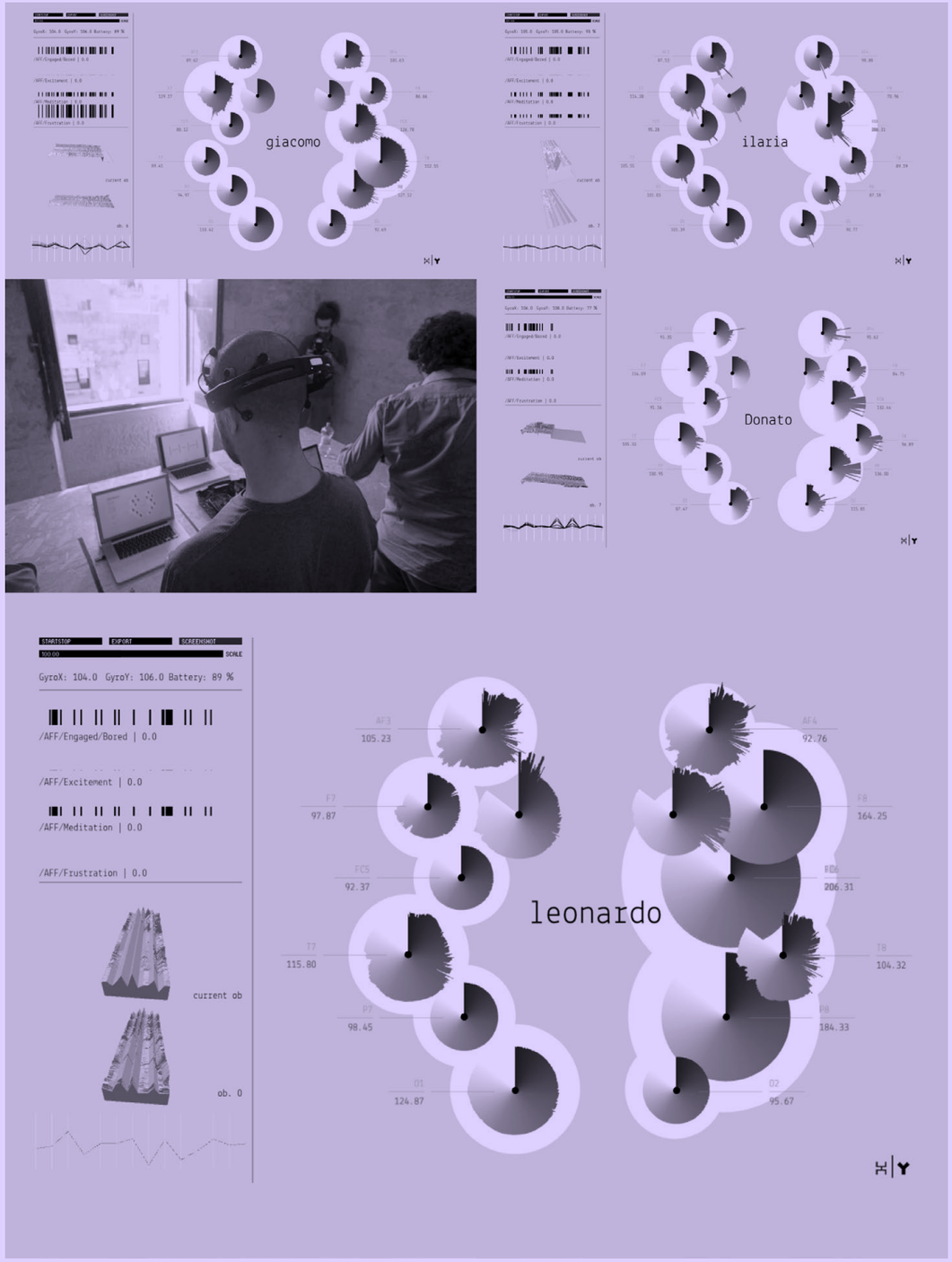






*Chocolate Chocolate Add Some Milk* was created by Kuan-Ju Wu in March 2010 as a real-time augmentation / interaction homework for the Special Topics in Interactive Art & Computational Design course taught by Golan Levin.

Kuan-Ju now teaches Creative Programming and Electronics at Jacobs Institute for Design Innovation at UC Berkeley and California College of the Arts.






# Morphology

A generative art project made with p5.js. Functions & geometries collectively give rise to various forms that waver between abstract & representational.

*Emily Xie, 2021*

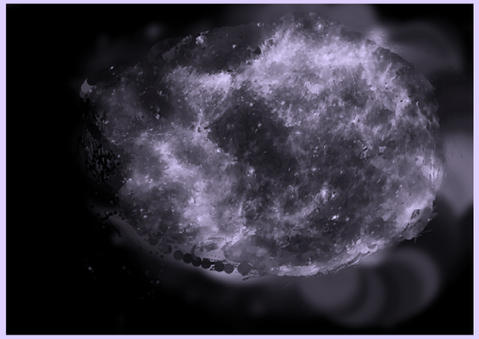
# @XRISPY

## CHRISTOPHER PHILLIPS
## UCLA DMA | 2010-14

*FOUNDER OF ART NOW CODING | artnowcoding.com*

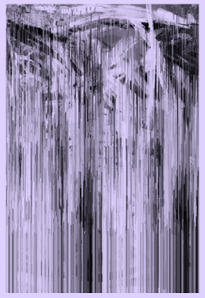
AUDIO PAINTINGS | 2021

Each pixel position and color is created from audio tracks + original color of photo



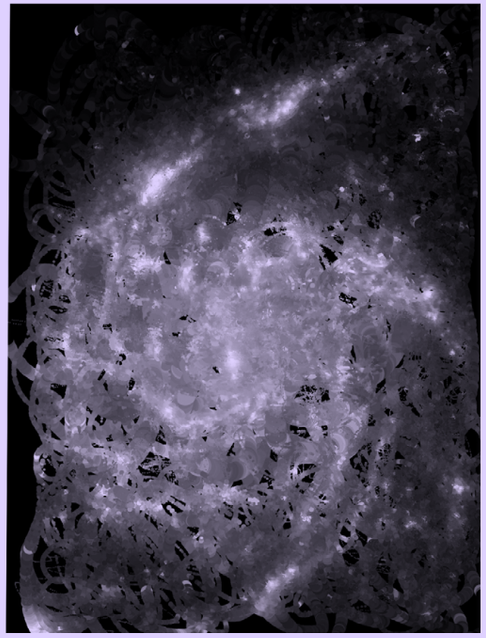
GALAXY BRUSH | 2021

Each color is referenced from Hubble Space Photographic Data, and then hand painted to product an realistic abstraction.

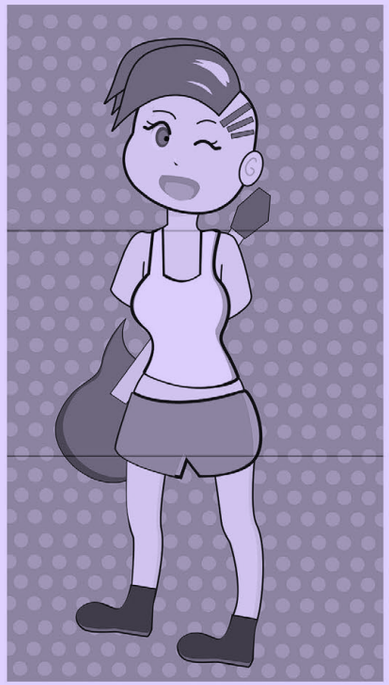


David Ian Brown Original hand painted abstract art + pixel Sorting technique

PIXEL GRAFFITI | *XRISPY + DAVID IAN BROWN*

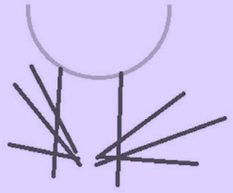






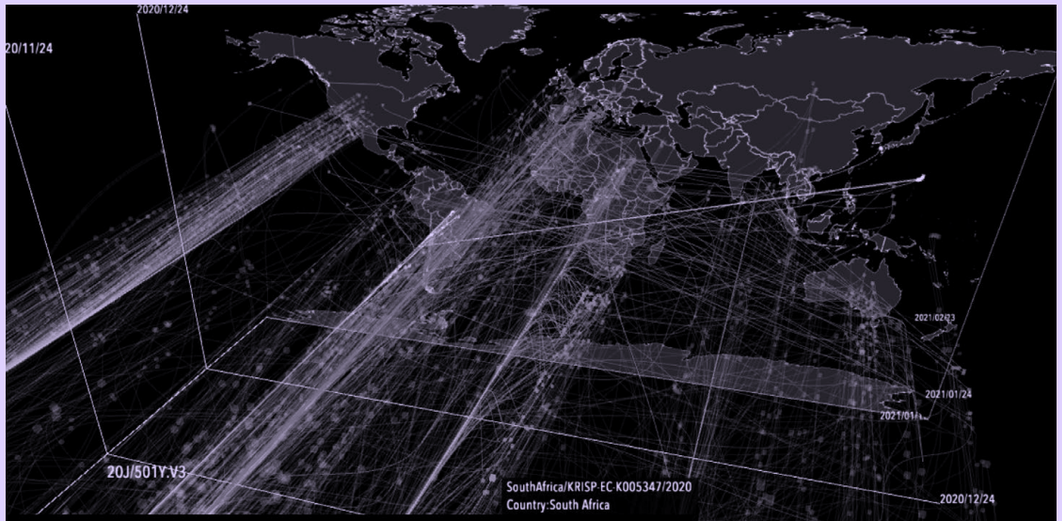
Original Full Body Sketch

Upper-Body Sketch



Lower-Body Sketch
by Yash Goyal

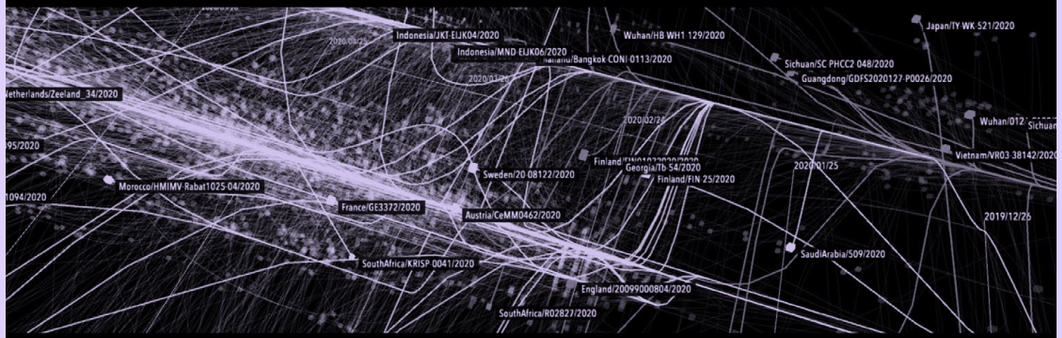
**Name:**
Shuyu Xu
**Work Description:**
It is interesting to combine different parts of stories into a new one. This project makes use of this funny method to create a surrealistic drawing to tell a fantastic story, which creates an unexpected experience about narrative. Three people will exchange the upper-body and lower-body, creating a new character's story of an exquisite corpse-like Frankenstein.
I combined three part of the body and draw them in p5js.
This is a punk girl who loves electronic music very much. She always takes her bass with her wherever she goes because she gets used to creating music immediately when she get some inspiration. This girl loves fashion but still not cares about her apparel very much. She believes that what makes herself good and comfortable is the most crucial thing in her life.

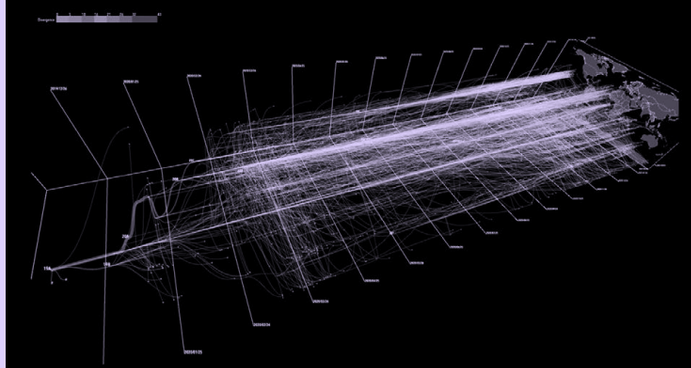



Scientific Visualization with Processing

## Time-space-based visualization of SARS-CoV-2 phylogeny

URL: https://timespacephylogeny.xyz

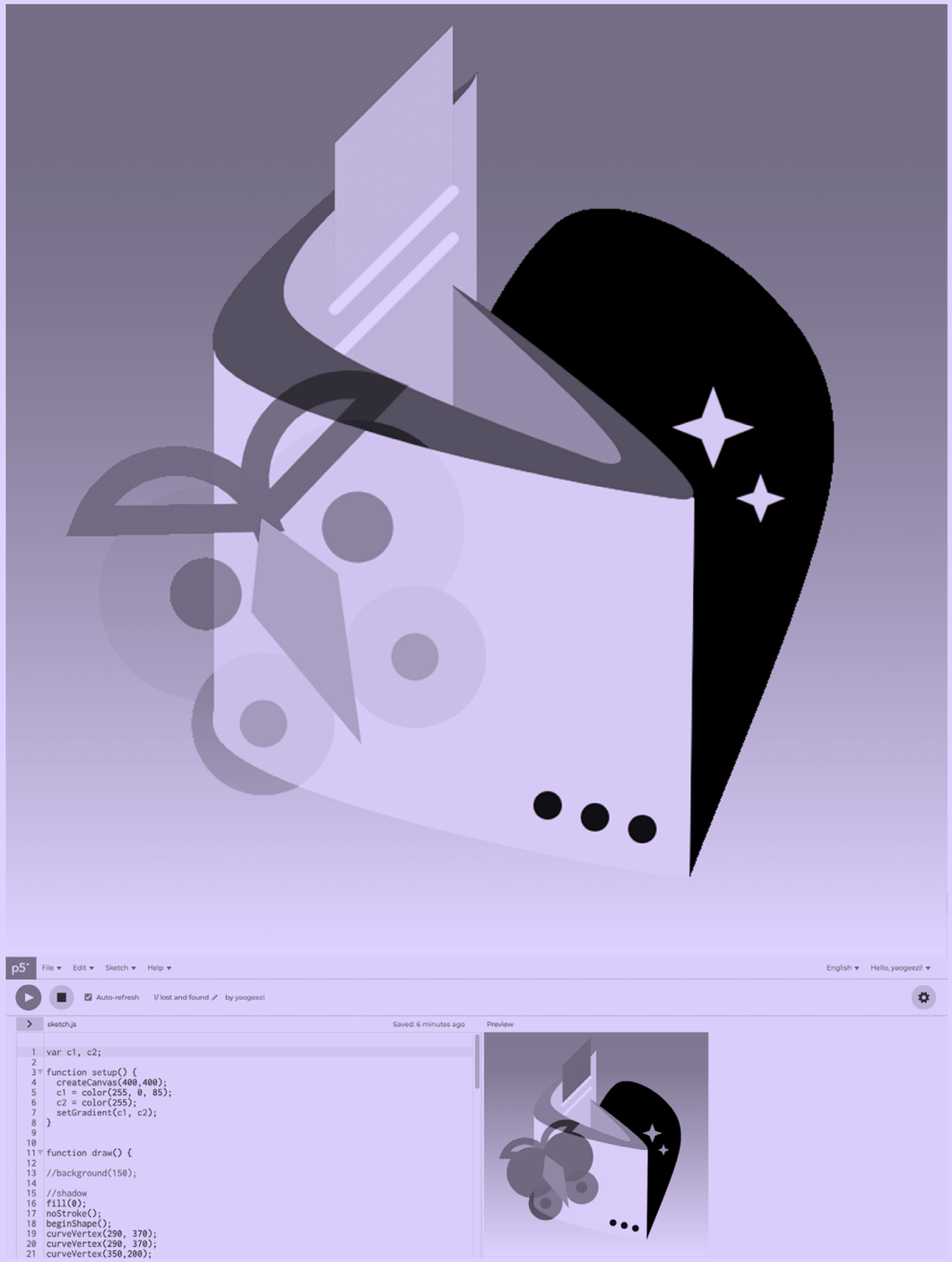


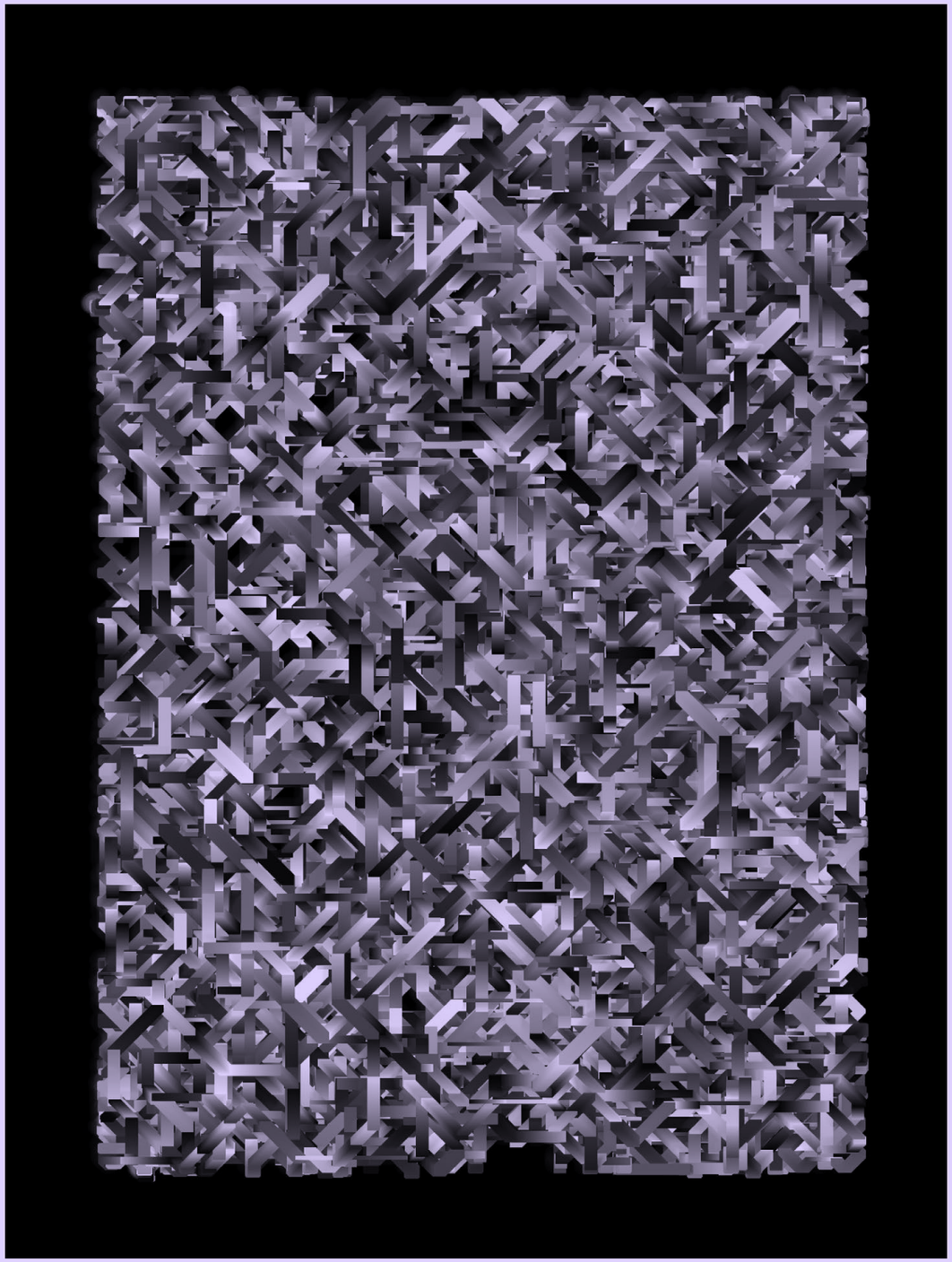



Some tools help in visualizing phylogenetic relationships of viruses based on their mutation patterns. However, merely associating phylogenies with time cannot enable full understanding of the geographical spread. Therefore, we proposed a method to associate phylogenies with both time and space, and combine them into a single three-dimensional graph.
We applied this method to visualize the phylogeny of SARS-CoV-2 causing COVID-19 to understand their temporal and geographic spread with Processing.
We subsequently generated a video to understand the spread of SARS-CoV-2 based on our system. Thre are more information and videos on our project website.

**Team:**
Masaki Yamabe, Visualization Design & Development
So Nakagawa, Scientific Advisor
Akira Wakita, Supervisor
Kentaro Oku, Technical Support
NHK Educational Corp., Scientific Research










PCD Shanghai 2019 Group Photo

PCD

# PROCESSING COMMUNITY DAY

# 社区日

February 24, 2019, 9:30AM - 4PM
2019年2月24日，上午9点半–下午4点

NYU Shanghai, Interacitve Media Arts
上海纽约大学，交互媒体艺术系

## Schedule 日程表

**Studio 826**

09:30 Welcome 欢迎

09:45 Introduction 简介

10:00 Show&Tell 快闪分享

- **Qianqian Ye:** Diversity & Inclusion in Processing Community
- **Yuli Cai:** Dance with Code
- **赵艺:** Fooling the Machine
- **吉乐米:** 看不见的界面
- **罗劬:** MinkeyFamily.com
- **徐亚茗:** Ghost in the Machine
- **张小佐:** 动态软装
- **周乐:** Creative AI, Collaborative AI and Chinese Heritage

12:00 Lunch Open Mic 午餐分享

14:00 Keynote 讲座

- **姚乐扬:** Wishlamp愿灯
- **罗宵:** Processing和微信
- **姜玥皞:** 山寨:硬件版的开源代码
- **UpteamCMI:** 新月
- **aaajiao:** .pde
- **茵梦:** 从数字艺术到数字文化

16:00 Closing & Group Photo
总结合照

**Studio 825**

**Workshop 工作坊**

11:00–12:00 | **Cici Liu**
Code it! Print it! with Processing

Lunch

13:00–14:00 | **Jung Hyun Moon**
Websketchbook+Machine Learning

14:00–15:00 | **Jiji Yu**
Glamorous wearables / 光华的衣服, LED穿戴装置。

**Studio 824**

**Workshop 工作坊**

11:00–12:00 | **Erin Zhang**
诗意测量装置，利用 p5.js 添加随机性。

Lunch

13:00–14:00 | **Leon Eckert**
Interactive geographic data with p5

14:00–15:00 | **Weijia Li**
RGB, 用P5.js 在浏览器中实现光的三原色并结合了Blob 元素的互动性小项目。

**Studio 818**

**Showcase 作品展示**

- **Prudence Yao:** Struggle
- **Jimmy Zhi:** "Are We Human?"
- **Siman Li:** 手相之诗
- **Yufeng Zhao:** Space Storm
- **Zeping Fei:** Generative Art Series: Possibility
- **姜玥皞:** 全网跨平台生态链2.0
- **UZKZ:** The Planet

https://processing-china.github.io

2019 PCD Shanghai 上海

Organized by Qianqian Ye, Chang Liu, and Yuli Cai




p5 for 50+
teaching

https://p5js.org/teach

English Español 簡体中文 한국어 हिंदी

p5.js

Home
Editor
Download
Donate
Get Started
Reference
Libraries
Learn
Teach
Examples
Books
Community
Showcase
Forum
GitHub
Twitter
Instagram

## Teach

Every teaching has its own unique goals, messages, conditions, and environments.By documenting and sharing p5 workshops, classes, and materials, we hope to better connect the p5.js learner and educator communities around the world.Share or recommend your own teaching experiences, too!

### p5 Teaching Resources

Search Filter →

Diversity & Inclusion : Gender | Race & Ethnicity | Language | Neuro-Type | Ability | Class | Religion | (Sub-)Culture | Political Opinion | Age | Skill Level | Occupation | #noCodeSnobs | #newKidLove | #unassumeCore | #BlackLivesMatter

Venue : Africa | Asia | Europe | North America | Oceania | South America | Virtual-Online

Year : -2014 | 2015 | 2016 | 2017 | 2018 | 2019 | 2020

Level of Difficulty : Elementary | Intermediate | Advanced

American Arts Incubator: Smarter Home (더 똑똑한 집)

*Lauren Lee McCarthy*

Venue & Date

Gwangju Cultural Foundation Gwangju, South Korea

2019 April 19~ May 9

Participants

artists, students, and any citizens residing in

Digital Weaving & Physical Computing Workshop Series

*Qianqian Ye & Evelyn Masso*

Venue & Date

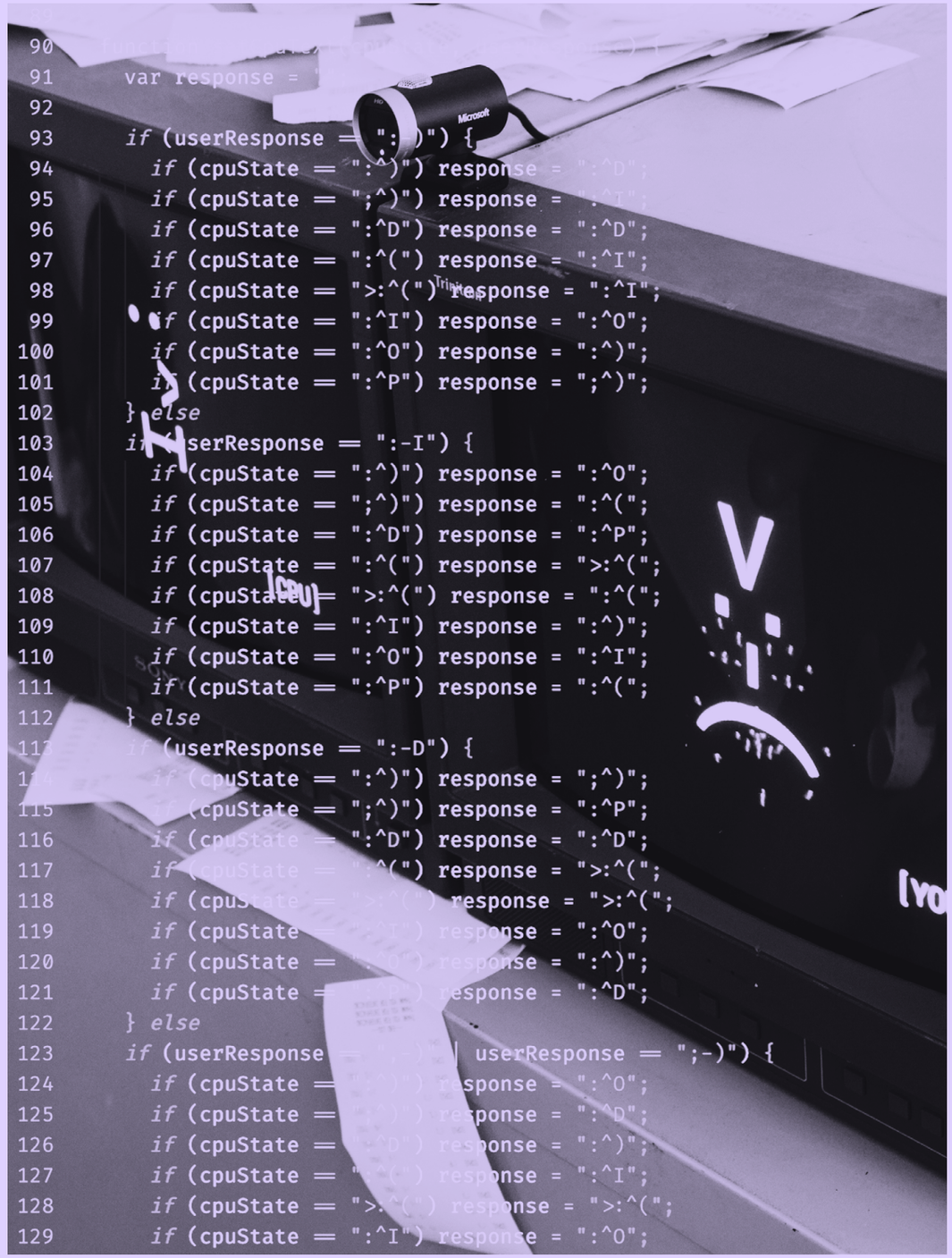
Womens Center for Creative Work (WCCW), Los Angeles, CA, US

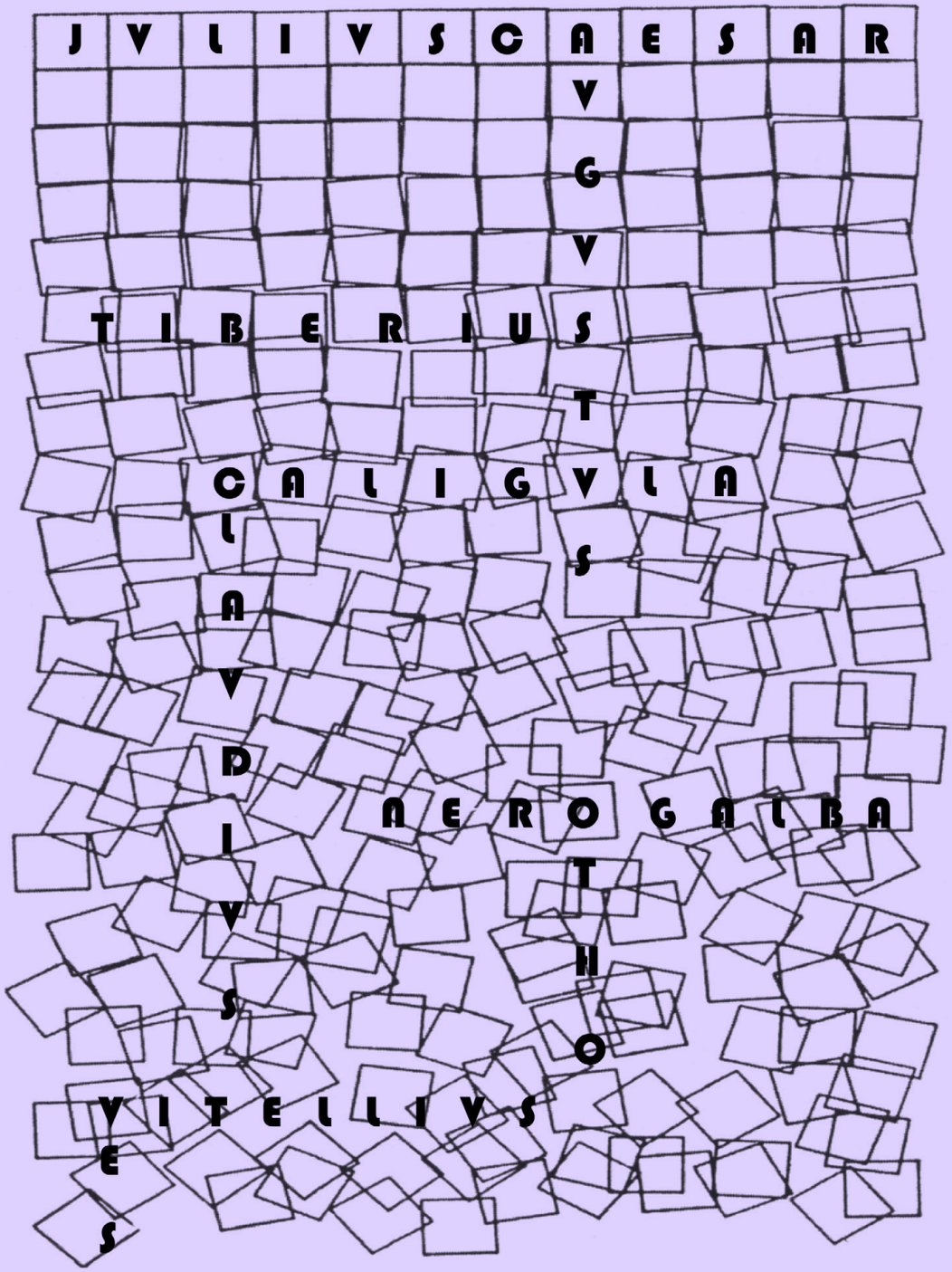
2019 October 19 - November 02, every Saturday 3-6 PM

Participants

15 women and non-binary artists, designer, makers, programers.



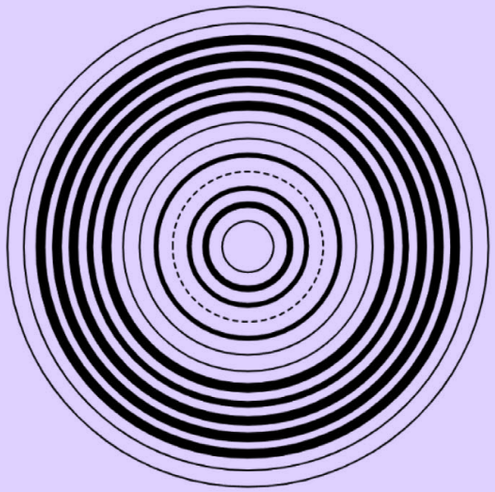



IVLIVSCAESAR
AVGVSTVS
TIBERIVS
CALIGVLA
CLAVDIVS
NEROGALBA
TITELLIVS
ES

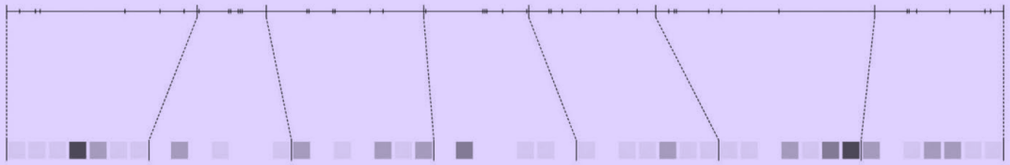


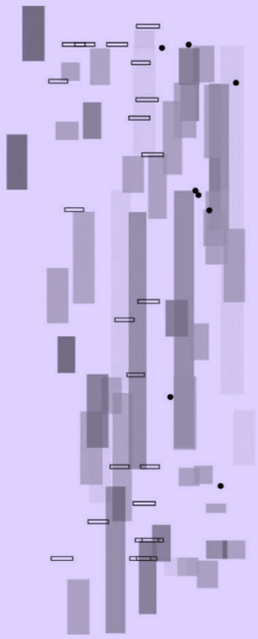

Aïda Diop
Pierre Boulez
Contrechamps
Di 07.02 2021
Ensemble Genevois
Contemporain et de Création
Salle
Frank Martin
contrechamps.ch
Rosenfeld
Seromoz
Contrechamps
Sa 13.03 2021
Ensemble Genevois
Contemporain et de Création
Centre d'Art
Contemporain
contrechamps.ch
Heinz Holliger
Contrechamps
Ma 10.11 2020
Ensemble Genevois
Contemporain et de Création
Studio
Ernest-Ansermet
contrechamps.ch
Kikoutchi
Menoud
Alice
Contrechamps
Sa 19.09 2020

Using p5.js for data sketching allows me to play with that data, making it both more meaningful (to me, by virtue of this expressive endeavor) and less meaningful (by using un-conventional visual mappings, or deliberately omitting some or all labels). In the process of *constructing these glyphs,* I can craft a familiarity or – hopefully- an intuition for the data.



*the resulting image is like the skin a snake sheds (an artifact of a process that extends beyond the artifact)*

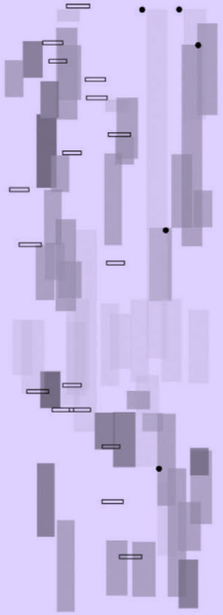




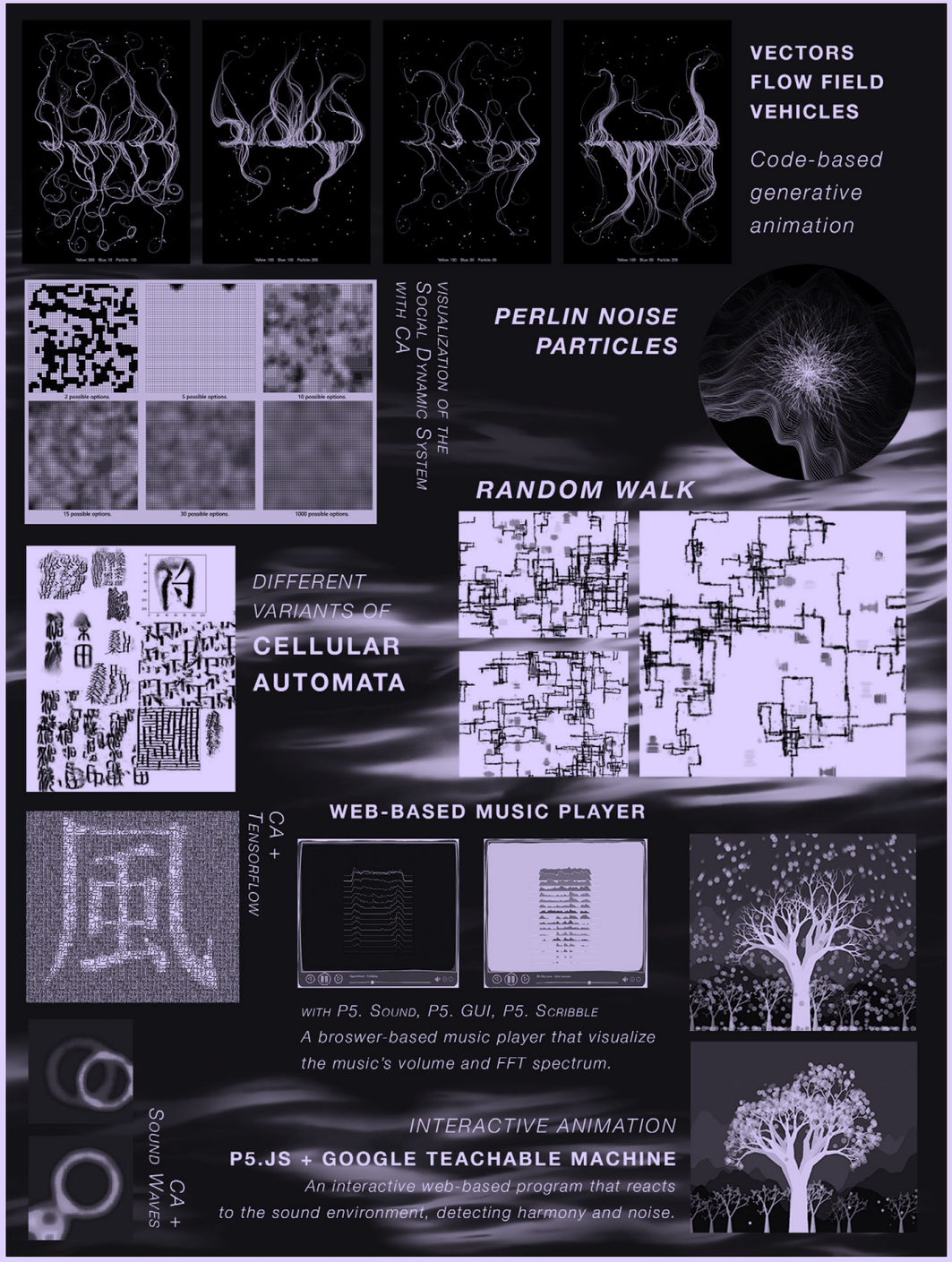
The passage of time sloughs data, leaks numbers and bits left and right. Animation and interaction, the way that size and shape and position can heave to show *change* is all part of how visual choices *construct time.*

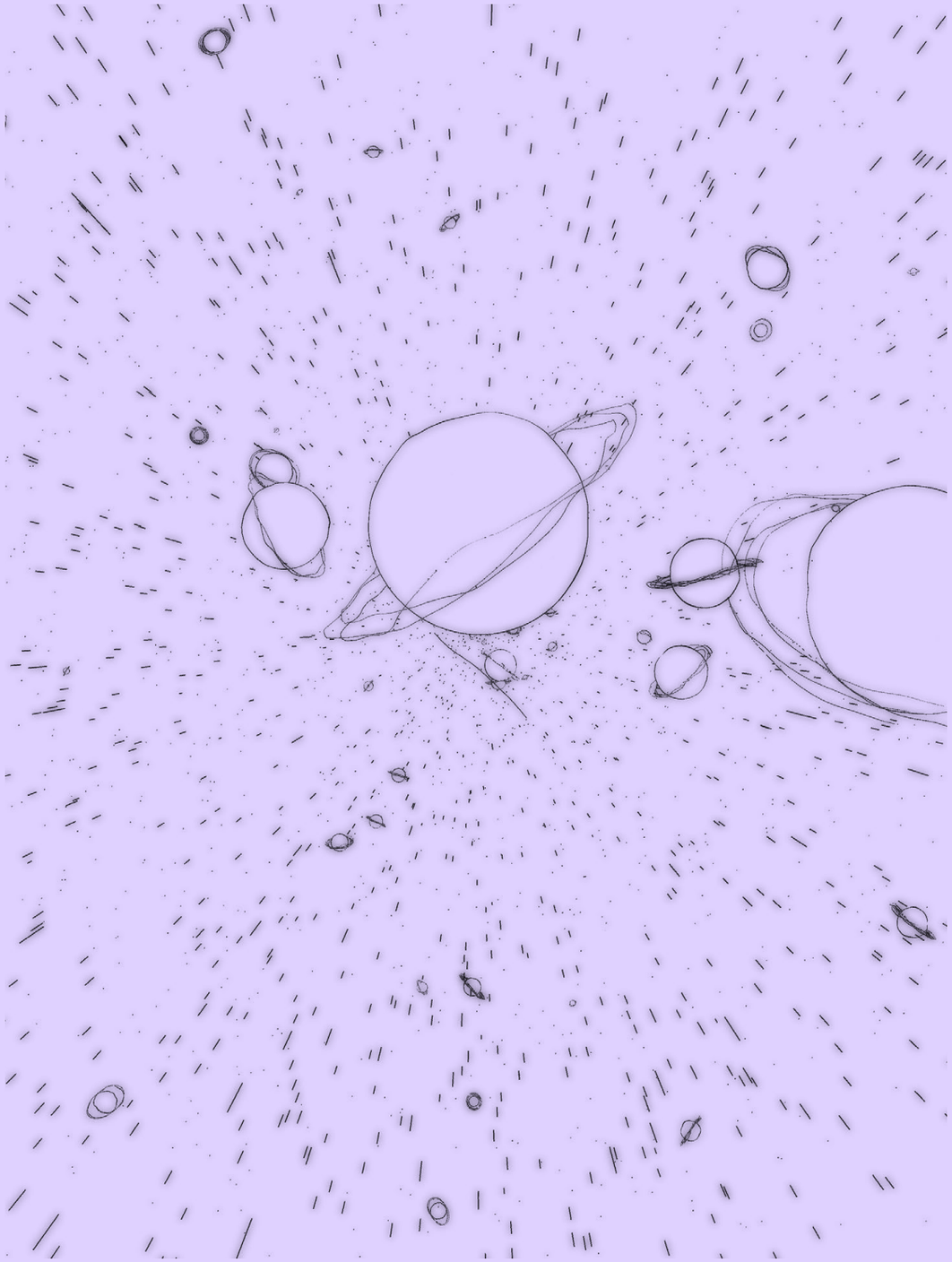
Randomness helps to *deconstruct false certainty*. The most usable tools for representing data visually take so many liberties. They seem to want to make everything look a lot more certain than it really is! The sharp edges and precisely-measurable sizes imply a false certainty. I don't want to flood the overall visual milieu with things that boast incommensurate certainty, but I want to make my tapestries of shapes and colors.

This is how I look at data that is difficult to look at too directly, with which I need familiarity and time.

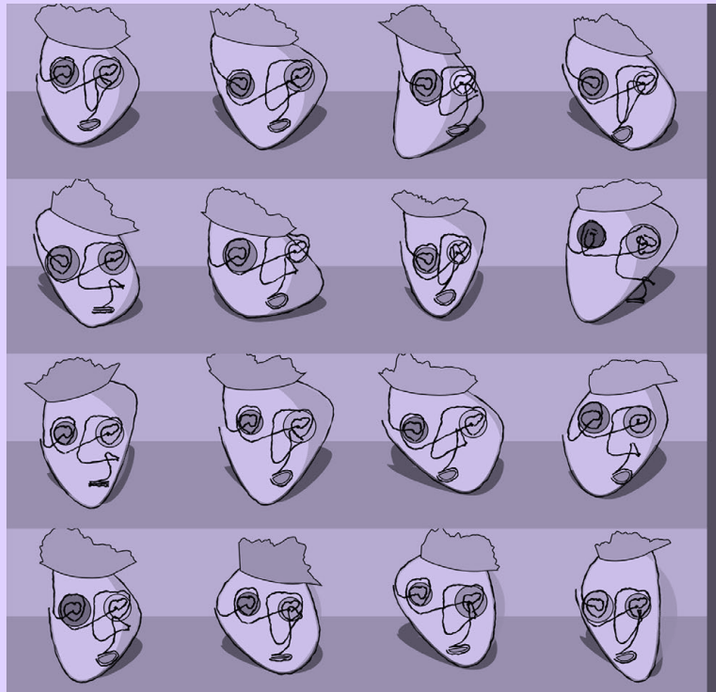


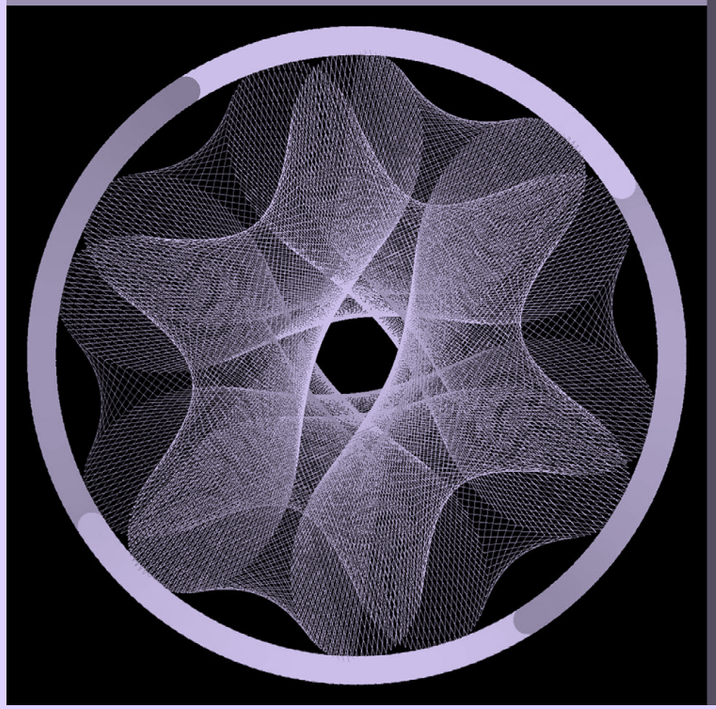













Made with P5JS

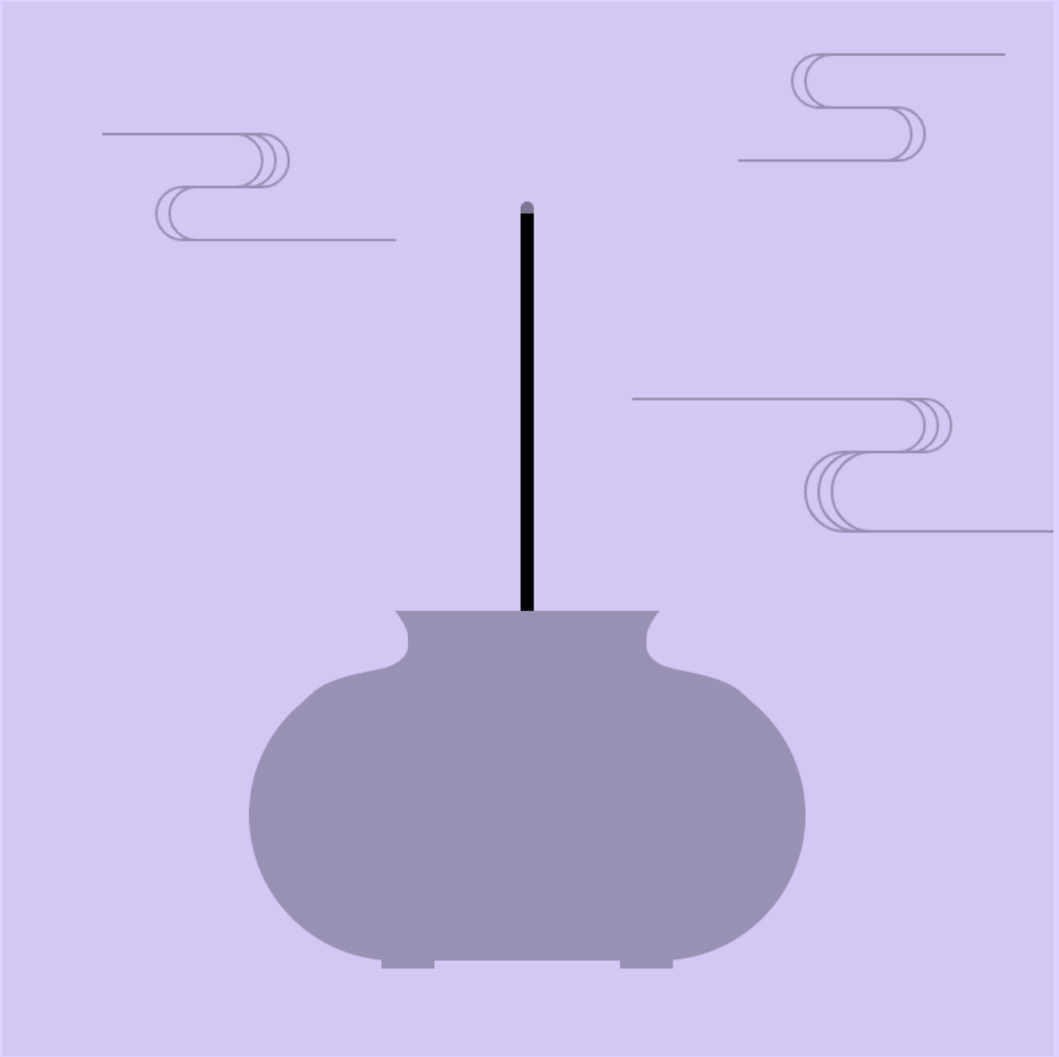
Made by Lingyi Zhou

Left Top:

Procedural Face Generator

Left Bottom:

Pattern Drew With Math



https://preview.p5js.org/yiqingzhou/present/2mMNU9PHv

## Incense Clock

In ancient China, incense was used as a timekeeper. An incense stick burns out indicates that 5 minutes have passed. Nowadays, we will still say "before the incense is burnt", which means "in 5 minutes".

In this project, I reproduced the incense timekeeper with p5.js. The incense stick slowly burns, and it takes 5 minutes for it to completely burns out. And the color pallet will change according to the time of day.



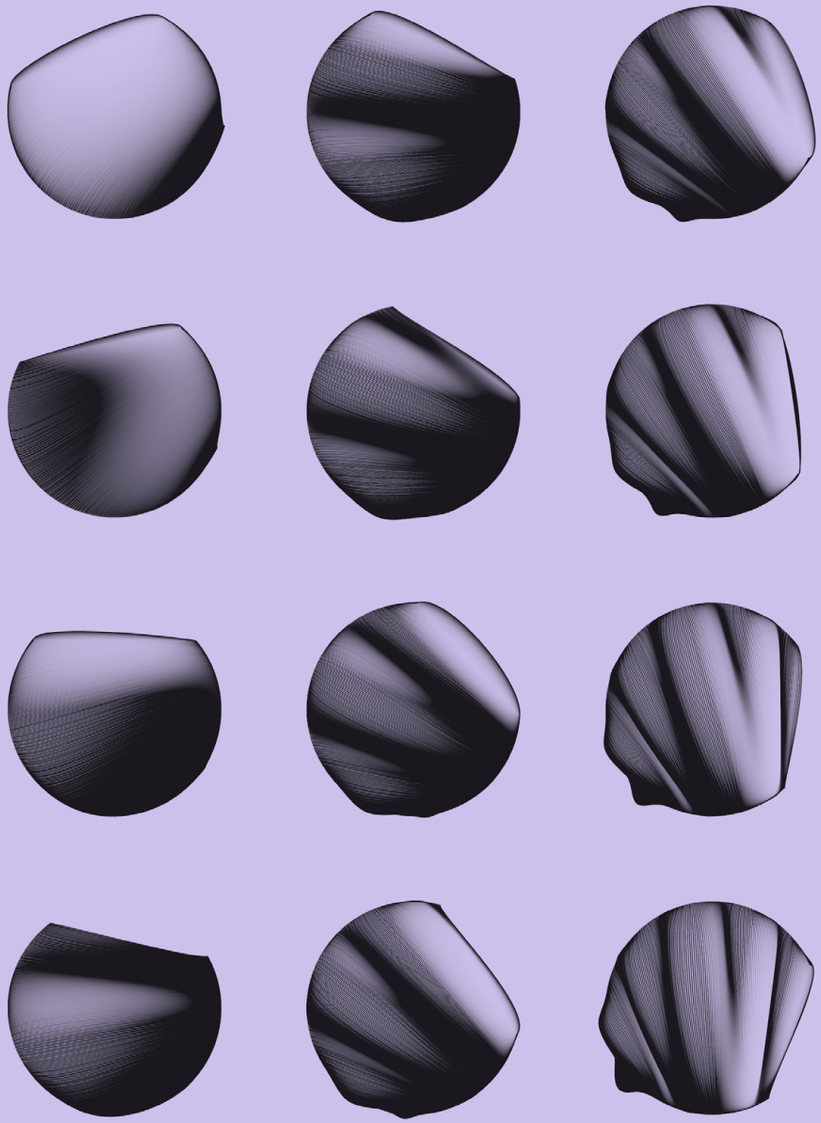



SpeckleSense exploits laser speckle sensing for precise, high-speed, low-latency motion tracking, which can be applied to a wide range of interaction scenarios and devices.

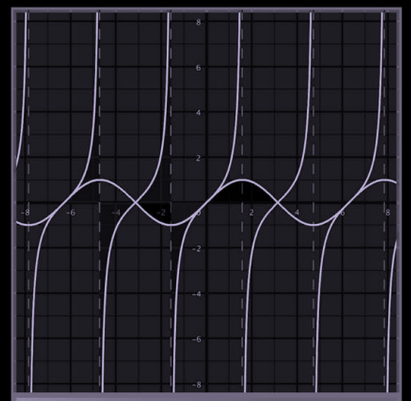
Jan Zizka Alex Olwal Ramesh Raskar olwal.com/specklesense

Sound reactive slit-scan remix of the 1987 film Wahnfried for the performance of DJ Spooky at Rifts & Scratches exhibition at Richard Wagner Museum Bayreuth in May 2017.
*Michael Zoellner*
*#emnullfuenf*
*Processing*

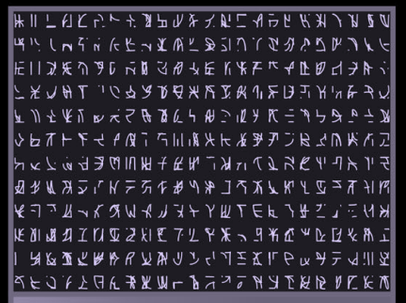




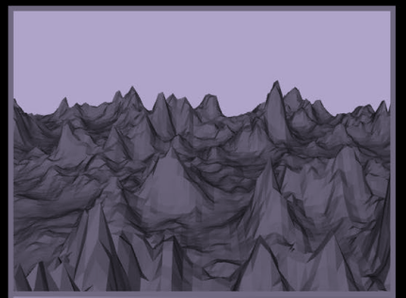
# nolan zurek



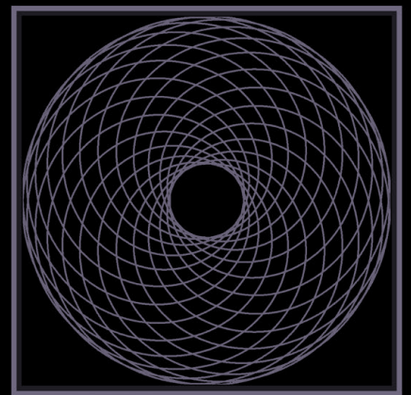
function plotter



glyph generator



3d terrain generator



spirograph renderer

painting generator

procedural terrain generator

github: nolanzurek

/*incantations for a radical future

Code is a lot like magic; put a few glyphs in the right order and you can summon almost anything.

we don't need capitalist logic.

"Learn to code" has become something of a cultural imperative, a way to futureproof oneself from professional obsolescence. It is the bootstrap by which communities are told to lift themselves up and improve their social and economic conditions.

As more women and people of color enter software development roles, the labor of coding is devalued. An increase in diverse hires has already led to an overall decrease in salaries, and the wage divide is as high as 20% between white male programmers and everyone else. Learning to code does not address structural inequities.

In a creative community, coding can be a powerful and liberatory practice. But when we only view code as a means of economic survival, it can never be empowering because labor will never free us.

we need community.

Community is more than bodies in a room or a connection shared through wire and glass.

Community is mutual care and sharing knowledge and making art and free childcare and wheelchair accessibility and a place to pump and mobility-aware seating and little kids talking during the presentations and vegan lunch options and brown skin and live

captions and online streaming and gender neutral bathrooms and si habla español and hair that looks like mine and asking what we can do for each other and everybody feeling good in this space we build together.

we need every contributor.

With software integrated into so many aspects of our public and private life, the distinctions between users and creators are collapsing. In open source, we depend on teachers, learners, artists, activists, developers, organizers, and volunteers of all sorts to sustain our work and community. Whether writing social codes or computer code, we are all contributors.

we need collaboration.

In Computer Science departments across higher education, to share one's code is to "cheat." Programs that measure software similarity and monitor students' keystrokes are used to catch code plagiarists. Even talking about a programming assignment can be seen as a violation and subject to punishment; solutions must be derived independently, or not at all.

In real life, repurposing old ideas to address new challenges happens all the time. Building on the work of others is necessary, celebrated, and efficient. We understand that most problems are better solved collectively; and yet, the architects of our increasingly software-dependent society are discouraged from working together.

I want to abandon the individualist, competitive model for teaching software development. Code should be shared, modified, remixed, revised, paraphrased, quoted, derived, copied, and pasted.

How far can we possibly go if everyone has to invent their own wheel?

we need a caretaking ethic.

As open-source contributors, we understand the value of sharing our work beyond the confines of proprietary restriction. But when our code can be weaponized against entire communities, our art can be minted and sold without our knowledge, and our intention of openness can be corrupted, exploited, and privatized to enrich the wealth of the morally bankrupt—we are faced with a conundrum.

Amidst the proliferation of NFTs, partnerships with ICE, and other moral concerns, open source communities must contend with the ethics of unfettered access. This is not merely a philosophical issue, but an existential one; when developers feel their work is misused, there is little recourse beyond removing their work from the public domain.

I want licensing models that are based on principles of consent. I want ongoing and revocable licenses that are granted enthusiastically. I know that people will say this kind of restriction is antithetical to the goals of open source, but I disagree. Black folks and Indigenous peoples know well the responsibility to protect our communities from economic and cultural

exploitation, and the consequences for not doing so. (Perhaps relatedly, there are even fewer of us in open source than in tech overall. Is that a tolerable restriction?)

There is a critical distinction between gatekeeping and caretaking; one gives limited access to a few, while the other ensures continued access for everyone. It's imperative that we recognize the difference.

*/ A.M. Darke

PER 001-02 021
Copyright Massachusetts Institute of Technology, courtesy Visible Language Workshop Archive, MIT Program in Art, Culture, and Technology (ACT).

PER 002-01 031
Alon Chitayat.

PER 002-03 038
Daniel Shiffman, The Coding Train.

PER 004-01 049
Alice Mira Chung, Content-Aware Collage. The images utilize repetition and image processing techniques to create absurd collages. The source images range from historic computer photographs to scientific illustrations of thermodynamic models, all topics extended around the concept of open-source collaboration.

PER 005-01 053
Alice Mira Chung.

PER 006-01 057
Taeyoon Choi.

PER 00701 059
Aren Davey.

PER 008-01 061
Alice Mira Chung.

PER 009-01 063
Alice Mira Chung.

PER 010-01 067
Alice Mira Chung.

PER 011-01 069
Alice Mira Chung.

PER 012-01 071
Alice Mira Chung.

PER 013-01 073
Israel Cedillo.

PER 013-02 073
Israel Cedillo.

PER 013-03 074
Israel Cedillo.

PER 013-04 075
Israel Cedillo.

PER 015-01 081
Alice Mira Chung.

EDU 101 136
School for Poetic Computation.

EDU 109 138
Saber Khan, Karen Blumberg.

EDU 110 138
Daniel Shiffman.

EDU 111 138
Maxwell Bigman, Casey Reas.

EDU 112 139
Ellen Nickels and Dominic Barrett.

EDU 113 139
Timmy Chen.

EDU 114 140
Ryan Gallagher and Eric Wild.

EDU 116 140
Colin Samuels and Ella Chung.

EDU 117 141
Ryan Gallagher.

EDU 118 141
Jolina Dominguez Kitaji Clément, Kelly Lougheed.

EDU 119 141
Yiting Liu.

EDU 120 142
STEMcoding Education Ohio, Kren Blumberg.

EDU 121 142
Saber Khan, John Umekubo, Dana Pompa.

EDU 122 142
Kelly Loughleed, Amjad Masad.

EDU 126 143
Sharon Lee De La Cruz.

EDU 127 143
Sharon Lee De La Cruz.

EDU 128 144
Aankit Patel.

EDU 129 144
Aankit Patel.

EDU 130 145
Kelly Loughleed.

EDU 131 145
Kelly Loughleed.

EDU 132 145
Lauren Lee McCarthy.

EDU 133 145
Ari Melenciano.

EDU 134 146
Sara Hendren.

EDU 135 146
Art Simon.

EDU 136 146
Melanie Hoff.

EDU 137 146
Tega Brain, Golan Levin.

COL 001 151
Motion Bank.

COL 002 151
voidLab.

COL 003 151
Feminist Center for Creative Work.

COL 004 151
Hillary Cleary, Christina Yglesias.

COL 005 152
Xin Xin, SHAWNÉ MICHAELAIN HOLLOWAY.

COL 006 153
Xiaowei R. Wang.

PRO 007 173
Chris Coleman.

PRO 014-01 203
Andrés Colubri.

PRO 014-02 203
Andrés Colubri.

PRO 014-03 204
Andrés Colubri.

PRO 015-01 206
Design Systems International.

PRO 015-02 206
Design Systems International.

PRO 015-03 206
Design Systems International.

PRO 015-04 206
Design Systems International.

PRO 015-05 206
Design Systems International.

PRO 015-06 206
Design Systems International.

PRO 015-07 206
Design Systems International.

PRO 015-08 206
Design Systems International.

PRO 015-09 206
Design Systems International.

PRO 015-10 206
Design Systems International.

PRO 015-11 206
Design Systems International.

PRO 015-12 206
Design Systems International.

PRO 015-13 206
Design Systems International.

PRO 015-14 206
Design Systems International.

PRO 015-15 206
Design Systems International.

PRO 015-16 206
Design Systems International.

PRO 015-17 206
Design Systems International.

PRO 015-18 210
Design Systems International.

P5* 001-01 215
Taeyoon Choi.

P5* 001-02 215
Taeyoon Choi.

P5* 001-03 215
Taeyoon Choi.

P5* 001-04 215
Taeyoon Choi.

P5* 001-05 215
Taeyoon Choi.

P5* 001-06 216
Taeyoon Choi.

P5* 002-01 219
Taeyoon Choi.

P5* 002-02 219
Taeyoon Choi.

P5* 002-03 219
Jess Klein.

P5* 004-01 224
Claire Kearney-Volpe, Luis Morales-Navarro, Mathura Govindarajan.

P5* 004-02 224
Claire Kearney-Volpe, Luis Morales-Navarro, Mathura Govindarajan.

P5* 004-03 224
Claire Kearney-Volpe, Luis Morales-Navarro, Mathura Govindarajan.

P5* 004-04 224
Claire Kearney-Volpe, Luis Morales-Navarro, Mathura Govindarajan.

P5* 004-05 224
Claire Kearney-Volpe, Luis Morales-Navarro, Mathura Govindarajan.

P5* 004-06 227
Claire Kearney-Volpe, Luis Morales-Navarro, Mathura Govindarajan.

P5* 005-01 235
Chelly Jin.

P5* 007-01 239
Jacquelyn Johnson.

P5* 007-02 239
Jacquelyn Johnson.

P5* 007-03 239
Jacquelyn Johnson.

P5* 007-04 239
Jacquelyn Johnson.

P5* 007-05 239
Jacquelyn Johnson.

P5* 007-06 239
Jacquelyn Johnson.

P5* 007-07 239
Jacquelyn Johnson.

P5* 007-08 239
Jacquelyn Johnson.

P5* 007-09 239
Jacquelyn Johnson.

P5* 007-10 239
Jacquelyn Johnson.

P5* 007-11 239
Jacquelyn Johnson.

P5* 007-12 239
Jacquelyn Johnson.

P5* 007-13 242
Jacquelyn Johnson.

P5* 008-01 243
Jacquelyn Johnson.

P5* 009-01 245
Aarón Montoya-Moraga.

P5* 009-02 246
Shawn Van Every, Jen Kagan, Tom Igoe.

P5* 009-04 247
Qianqian Ye.

P5* 009-05 247 Manaswini Das, Nancy Chauhan, Shaharyar Shamshi.

P5* 009-07 248 Layla Quiñones, CS4ALL.

P5* 009-10 249 Kate Hollenbach.

COM 002 270 Danielle Freiman, Olivia Glennon, Rebecca (TK).

COM 003 271 Israel Cedillo.

COM 004 272 Nicolás Troncoso López, PCD Santiago de Chile.

COM 004 272 Sergio Venancio, PCD São Paulo.

COM 004 272 PCD India.

COM 004 272 Becca Rose, PCD Bristol.

COM 004 272 Becca Rose, PCD Bristol.

COM 004 272 Winnie Soon, PCD Aarhus.

COM 004 273 PCD Ethiopia.

COM 004 273 PCD India.

COM 004 274 Lee Tusman, PCD NYC.

COM 004 274 Sergio Venancio, PCD São Paulo.

COM 004 274 Aarón Montoya-Moraga, PCD Quito.

COM 004 274 Nicolás Troncoso López, PCD Santiago de Chile.

COM 004 274 PCD Porto.

COM 004 274 Aarón Montoya-Moraga, PCD Quito.

COM 004 274 Marlus Araujo, PCD Rio de Janeiro.

COM 004 274 PCD Brasil.

COM 004 275 Nicolás Troncoso López, PCD Santiago de Chile.

COM 004 275 Qianqian Ye, PCD Shanghai.

COM 004 275 PCD Coimbra.

COM 004 275 PCD Japan.

CON 001 316 Instagram: @12niki14.

CON 002 317 @a.mehow.c.

CON 003 318 Rozina Aamir.

CON 004 319 @aamirshaikh95.

CON 005 320 “Hubbub”, by Angelabelle Abarientos, IG: @0.1.a.b.

CON 006 321 @aduarte.io.

CON 007 322 Tanve Agerwal.

CON 008 323 Arianna Agudio, Communication and Interaction Designer.

CON 009 324 Memo Akten, @ memotv.

CON 010 325 Hind Al Saad @ hindgalsaad.

CON 011 326 joshuaalbers.com.

CON 012 327 Ale @414c45.

CON 013 328 algorat.club, @alg0rat.

CON 014 329 Christo Allegra / christoallegra.com.

CON 015 330 Juan Alonso (@kokuma).

CON 016 331 https://www.instagram.com/alptugan/.

CON 017 332 @amandaghassaei.

CON 018 333 @anderkoy.

CON 019 334 @andor_saga.

CON 020 335 Alexandre Andrada - Title: man += machine - @aand.lab.

CON 021 336 Kyle Ang, kyleang.com.

CON 022 337 Anushka aka opheliagame.

CON 023 338 Name: Anxious to Make (Liat Berdugo + Emily Martinez) Title: Confidence Generator Link: https://anxioustomake.ga/confidencegenerator.

CON 024 339 Aranda\Lasch, @ arandalasch.

CON 025 340 Name: Aranya (Ritesh Lala), Twitter: @ zenaranya.

CON 026 341 @ArjanBytez.

CON 027 342 Lolo Armdz (@armdz).

CON 028 343 Joey Aronson, www.arons0n.com.

CON 029 344 Art Blocks.

CON 030 345 Kim Asendorf.

CON 031 346 Valentine Cat, created 14th February 2015 by Rain Ashford.

CON 032 347 Copyright: Jorge Ayala + Alessio Erioli Photo & Model: Atarah Atkinson Year: 2014.

CON 033 348 @aymdes, Aymeric Descamps, designer.

CON 034 349 Mitul S Ayyod (Instagram - @ mitulayyod).

CON 035 350 @b2renger.

CON 036 351 @b3aribeiro.

CON 037 352 Ari Bader-Natal.

CON 038 353 Justin Bakse compform.net.

CON 039 354 Bert Balcaen, Corneel Cannaerts and Jan Vantomme.

CON 040 355 Giorgia Bandiera & Maria Alessandra Fratta.

CON 041 356 Giorgia Bandiera, Visual Designer, behance> giorgiabandiera @ knockthedot.

CON 042 357 Lali Barrière, lalibarriere.net

CON 043 358 xavier barriga abril · @ runamora.

CON 044 359 Beardcoded.

CON 045 360 M. James Becker @ mjamesbecker.

CON 046 361 George Benainous, CS Educator.

CON 047 362 Frank Benkelmann | Twitter: @benkelmann.

CON 048 363 Beta HSU.

CON 049 364 Mrityunjay Bhardwaj, Head of AI, mrityunjay.ml.

CON 050 365 Nicola Bischof.

CON 051 366 Joëlle Bitton.

CON 052 367 Jaap Blonk.

CON 053 368 name: Jonas Bo Twitter: @k4brio Insta: @j0nasbo year: 2009.

CON 054 369 Eleonora Bonorva, Nicola Bischof, Instagram of Eleonora: @eliiiifant.

CON 055 370 Jay Borgwardt, @jay.borgwardt.

CON 056 371 David Bouchard (https://deadpixel.ca).

CON 057 372 Claudia Brambilla.

CON 058 373 Jono Brandel.

CON 059 374 Fred Briolet.

CON 060 375 Mike Brondbjerg (@ mikebrondbjerg / www.kultur.design).

CON 061 376 Ashley James Brown.

CON 062 377 Buffalo Geological Consulting buffalo@ buffalogeo.com.

CON 063 378 p5.js illustration by Erika Bulger (erikabulger.com), description by Patricia Huang.

CON 064 379 bustavo.com.

CON 065 380 Cacheflowe / Justin Gitlin.

CON 066 381 Alfredo Calosci - negot.net.

CON 067 382 https://www.instagram.com/camillerouxart/.

CON 068 383 camilosandoval.de.

CON 069 384 A project by Carlson Garcia (Esteban Garcia Bravo, Max Carlson and Aaron Zernack).

CON 070 385 Anna Carreras @ carreras_anna www.annacarreras.com.

CON 071 386 Rodrigo Carvalho | @ visiophone_lab.

CON 072 387 Guillermo Casado. Peripecio. Penrose layouts.

CON 073 388 Diego Cataldo - Screenprint - Untitled (2021).

CON 074 389 Ricardo Cedeño Montaña. @drnn1076.

CON 075 390 Chan Jun Shern (@ junshern on GitHub).

CON 076 391 Katie Chan.

CON 077 392 Angela Chang, Github: https://github.com/anjchang, Twitter: https://twitter.com/anjchang.

CON 078 393 Anny Chang.

CON 079 394 Michael Chang (@ mflux).

CON 080 395 Tingwu Chang.

CON 081 396 @CharStiles.

CON 082 397 Jose Chavarría / Instagram: @chavis.el.bueno.

CON 083 398 Han Chen.

CON 084 399 Face Generator - Kayla Chen | MFA Design & Technology / Parsons School of Design Critical | Computation Lecture + Lab | https://cclab-kaylachen-portfolio.glitch.me | cheny184@newschool.edu | IG: ys.kayla.

CON 085 400 Georgios Cherouvim - https://ch3.gr.

CON 086 401 Hye Min Cho, hyeminchocho.com.

CON 087 402 CHRIST L - PARSONS MFADT 23'.

CON 088 403 @chrleon.

CON 089 404 alm chung (@ nekofutura).

CON 090 405 Citizen Sticks: Emily Bright, Garrett Beleu, Joel Slayton, Steve Durie.

CON 091 406 John Clavin.

CON 092 407 Co-de-iT (team: Tommaso Casucci, Michele Semeghini) Year: 2014 Link: https://www.co-de-it.com/p5experimental-3d-printing.html.

CON 093 408 cocopon.

CON 094 409 code & share [ ], Aarhus, www.pcdaarhus.net, IG: codeshareaarhus.

CON 095 410 +CODE Cultura Digital - pluscode.cc - @ pluscodefest.

CON 096 411 Communication Design / Hof University / Campus Muenchberg.

CON 097 412 Casey Conchinha https://kccon.ch/, Louise Lessél http://www.louiselessel.com/.

CON 098 413 Michael Connolly.

CON 099 414 Stefano Contiero // stefan_contiero.

CON 100 415 Alba G. Corral.

CON 101 416 Akio Correll.

CON 102 417 Webpage: https://cosmicbhejafry.github.io/.

CON 103 418 Kaue Costa // kaox.tv.

CON 104 419 Posters created to announce the Creative Code Stammtisch monthly meetup in Berlin by various community members, many of them using Processing. https://creativecodeberlin.github.io/Stammtisch/.

CON 105 420 Creative Coding Amsterdam https://twitter.com/cc_amsterdam.

CON 106 421 Creative Coding Madrid.

CON 107 422 The community of creativecodingutrecht.nl.

CON 108 423 The community of creativecodingutrecht.nl.

CON 109 424 The community of creativecodingutrecht.nl.

CON 110 425 @ creativecodecollective.

CON 111 426 CreativeCodingCircle ARTiS.

CON 112 427 Jeanne Criscola.

CON 113 428 Sterling Crispin.

CON 114 429 CS4ALL.

CON 115 430 CS4All NYC (Twitter: CSforAllNYC, website: https://www.blueprint.cs4all.nyc/).

CON 116 431 James Curran @ minusminusmusic.

CON 117 432 @cyberia.

CON 118 433 czausner.com.

CON 119 434 Name - Ankita D'Souza Instagram- @ onecuriousdsouza.

CON 120 435 Luca Damasco, Github: Luxapodular.

CON 121 436 DanDan_DCA.

CON 122 437 Danie'Lopes.

CON 123 438 Manaswini Das (GitHub: manaswinidas).

CON 124 439 Rupak Das.

CON 125 440 DataFlaw - https://www.liinks.co/dataflaw.

CON 126 441 Aren Davey (Twitter: @_aahdee_)

CON 127 442 Erica David, IG @erica.david230.

CON 128 443 Jeff Davis.

CON 129 444 Joshua Davis / PrayStation.

CON 130 445 Ted Davis, GH: @ffd8, IG: @teddavisdotorg.

CON 131 446 Brendan Dawes.

CON 132 447 dcuartielles.

CON 133 448 Flor de Fuego | ig: @flordefuega | https://flordefuego.github.io/ | tw: @flordefuego4.

CON 134 449 @deconbatch.

CON 135 450 Fanny Demier.

CON 136 451 Fanny Demier and Kajetan Som.

CON 137 452 Fabian Dennler.

CON 138 453 Matt DesLauriers (mattdesl).

CON 139 454 DigitalColeman.

CON 140 455 rafa diniz - creative code educator and artist | @rafaddiniz @ codigos_fazem_arte.

CON 141 456 @diogo_navarro.

CON 142 457 Andrea Diotallevi.

CON 143 458 Victor Doval.

CON 144 459 Dr. Woohoo!

CON 145 460 R. Luke DuBois.

CON 146 461 Karan Dudeja.

CON 147 462 Gabriel Labov Dunne.

CON 148 463 Steve Durie, CADRE Media Lab, SJSU.

CON 149 464 E.C.H.

CON 150 465 @eddietree.

CON 151 466 Kelly Egan https://kellyegan.net.

CON 152 467 Miguel Elizalde / @ urbaninfrasound.

CON 153 468 Ursula Endlicher, www.ursenal.net.

CON 154 469 Equipo eQuills: Andres Ochoa, Andrew Alva, Eli Ramos & Juan Fuentes. www.equills.letrismo.com.

CON 155 470 Mark Ericson, @m_cericson.

CON 156 471 JM Escalante.

CON 157 472 ethan_omo on Instagram.

CON 158 473 Brian Evans.

CON 159 474 evhan55.

CON 160 475 Generative Poetry by Rafael Fajardo.

CON 161 476 FAL Works.

CON 162 477 Behnaz Farahi. Designer, Artist, Critical maker. www.instagram.com/behnazfarahi/.

CON 163 478 Nahum Farchi.

CON 164 479 instagram.com/feamonkey.

CON 165 480 Nicholas Felton.

CON 166 481 Yu-Ting Feng.

CON 167 482 German Fernandez Cantos.

CON 168 483 Rodrigo Fernández Flores (@ templodespejos).

CON 169 484 Luis Ferreira (@schuur.creations).

CON 170 485 Nick Fox-Gieg.

CON 171 486 Luis E. Fraguada.

CON 172 487 F#READY, GitHub: https://github.com/FreddyOffenga Demozoo: https://demozoo.org/sceners/35273/.

CON 173 488 Saskia Freeke.

CON 174 489 Jeff French, jwfrench.com.

CON 175 490 Max Frischknecht, https://maxfrischknecht.ch/.

CON 176 491 @frwrdnet.

CON 177 492 Masaru Fujii / @ozachou_g.

CON 178 493 @functiondraw.

CON 179 494 Julien Gachadoat | @v3ga.

CON 180 495 Sergio Galan @sergioeclectico (twitter), Victor Diaz @victordiaz (github).

CON 181 496 https://games.qu.edu/.

CON 182 497 gammastop.

CON 183 498 Ananya Ganesh.

CON 184 499 Title: XV. by Esteban Garcia Bravo (@snebtor) in collaboration with Aaron Zernack (@wound_design) and Jorge Garcia.

CON 185 500 Sofia Garcia | @sofiagarcia_io.

CON 186 501 @garrettbeleu Artist.

CON 187 502 https://www.instagram.com/genetypo77/.

CON 188 503 gerard_sgs Twitter.

CON 189 504 josh giesbrecht.

CON 190 505 gin_graphic.

CON 191 506 giy (https://gangilyi.xyz/), Spirals - https://github.com/morfant/pglr/tree/master/spirals_spirals.

CON 192 507 Zahra Golestanha.

CON 193 508 Amy Goodchild.

CON 194 509 Twitter: @gorillasu - Blog: gorillasun.de.

CON 195 510 Title: Movement / Author: Gotutiyan.

CON 196 511 Vadim Gouskov insta: @vadim.gouskov.

CON 197 512 Ryan Govostes.

CON 198 513 Gramener. Twitter : @Gramener.

CON 199 514 daniel grantham.

CON 200 515 twitter: @grasser_alex ; homepage: alexandergrasser.com.

CON 201 516 Ira Greenberg.

CON 202 517 Stalgia Grigg.

CON 203 518 Sarah Groff Hennigh-Palermo.

CON 204 519 Benedikt Groß, Hartmut Bohnacker, Julia Laub, Claudius Lazzeroni, Joey Lee, Niels Poldervaart.

CON 205 520 @grunskm.

CON 206 521 Nuno Guedes Silva.

CON 207 522 Bilge Günay (Dist Collective).

CON 208 523 Mario Guzman, @marioguzzzman, https://www.mario-guzman.com/.

CON 209 524 Andreas Gysin & Sidi Vanetti.

CON 210 525 Peter Ha | Instagram: @hiddenenigma | Project: 365 Processing, 2014 | https://peterha.net/projects/365-Processing/.

CON 211 526 Hanif Haghtalab, Noise Field Study, https://www.instagram.com/hanif.hb/.

CON 212 527 Gottfried Haider.

CON 213 528 Xiaoxu Han, https://xiaoxu-portfolio.glitch.me/.

CON 214 529 Oliver Hanstein (Nodeart).

CON 215 530 Usman Haque.

CON 216 531 Alex Harding.

CON 217 532 Connor Hasting.

CON 218 533 Fabian Heller, Instagram: @farnfreund.

CON 219 534 Daniel Giles Helm.

CON 220 535 @henry_null_.

CON 221 536 Caroline Hermans.

CON 222 537 Leander Herzog @lennyjpg.

CON 223 538 Julian-Anthony Hespenheide (ig: julian_hespenheide / git: https://github.com/ndsh / www.julian-h.de ).

CON 224 539 the weight of light (2015), Martin Hesselmeier & Andreas Muxel, physics engine (P5) by Micha Thies.

CON 225 540 Hevey (@HeveyArt - https://hevey.art).

CON 226 541 Tyler Hobbs.

CON 227 542 Robert Hodgin.

CON 228 543 hoehoe, Buddha Machine Mirage, TW:@hoehoe.

CON 229 544 Jona Hoier.

CON 230 545 Kate Hollenbach / @kjhollen / katehollenbach.com.

CON 231 546 Michael Honey (twitter: @michaelhoney).

CON 232 547 HosodaMath.

CON 233 548 Jiangnan Hou, IG: @jiangnan_nichole.

CON 234 549 John Houck.

CON 235 550 Toby Howard @tobyhoward.

CON 236 551 https://huaigulin.github.io/.

CON 237 552 nick hubben/@klaushubben.

CON 238 553 hubstruct.

CON 239 554 Chris Hunt and Ben Dunks.

CON 240 555 @hypereikon.

CON 241 556 I've made up my mind to develop a new frontier of art with Processing.

CON 242 557 Ilgın İçözü (Dist Collective).

CON 243 558 Lisa Ihde, GitHub: @julisa99, website: lisaih.de.

CON 244 559 Tom Igoe.

CON 245 560 Psychedelic eye by ilithya - @ilithya_rocks.

CON 246 561 incre.ment.

CON 247 562 Indigemoji.

CON 248 563 Codice Inutile, www.codiceinutile.org, @codiceinutile.

CON 249 564 @isabelazamith.

CON 250 565 Marcinia Jacquet @marsenner.

CON 251 566 Mahima Jain (@mahima_j).

CON 252 567 Jbarbeau.art.

CON 253 568 Jeff (ippsketch).

CON 254 569 Jeffrey - dodely.dev.

CON 255 570 Florian Jenett.

CON 256 571 Lazar Jeremic, lazar.ch.

CON 257 572 Yuehao Jiang, Artist based in Los Angeles, IG: yuehao_jiang.

CON 258 573 @jmutterer for Colmar Basket.

CON 259 574 @jnsdbr.

CON 260 575 @johnbcarpenter // http://johnbcarpenter.com.

CON 261 576 Jerel Johnson.

CON 262 577 Joshua Marris.

CON 263 578 Jucarvidal, my name is Juan Carlos vidal, I am a fullstack developer but I lean more towards the frontend, and I love interactivity and generating experiences I am Colombian, my twitter is @jcvmarin, my instagram is @jcvidal81.

CON 264 579 twitter, instagram: @julian__weis.

CON 265 580 @Julien_Espagnon.

CON 266 581 Jun, Insta: void_draw_the_future.

CON 267 582 @JunieGenius.

CON 268 583 Rodrigo Junqueira (Rodjun) - github.com/rodjuncode.

CON 269 584 Verginiya Kadina, @g.i.n.i.y.a.

CON 270 585 maja kalogera.

CON 271 586 Instagram Handle : @azz_adi Name : Aditya Kanade.

CON 272 587 Ashley Kang / ashleykang.dev.

CON 273 588 Indhu Kanth, @being_ink (Instagram).

CON 274 589 kasperkamperman.com (@kasperkamperman).

CON 275 590 Roni Kaufman.

CON 276 591 website: herry.kim.

CON 277 592 Joy Juyeon Kim.

CON 278 593 MiHyun Kim, Assistant Professor of Communication Design at Texas State University. https://www.instagram.com/mihyun____kim/.

CON 279 594 Name: Yonsan Kim, Social Media Handle: @yonsankm.

CON 280 595 Kitsunebi, @Dai_Hanabi on Twitter.

CON 281 596 instagram: kleemotfd, github: kcarollee.

CON 282 597 Aaron Koblin.

CON 283 598 Aaron Koblin and Daniel Massey.

CON 284 599 Osman Koc (@kocosman).

CON 285 600 Koh Achim, Data Editor at Alookso. @achimkoh.

CON 286 601 Tetsu Kondo, Tetraleaf.

CON 287 602 Tetsu Kondo, Processing Japanese Translation.

CON 288 603 Jules Kris www.juleskris.com.

CON 289 604 Jan Kubasiewicz & Scott Murray.

CON 290 605 Rupesh Kumar (iamrupesh.me).

CON 291 606 Kumu Jacqueline & Kumu Tiana.

CON 292 607 kusakari (Twitter @kusakarism, OpenProcessing https://openprocessing.org/user/224308).

CON 293 608 LadyKStudios.

CON 294 609 Laleh Mehran & Chris Coleman.

CON 295 610 Sam Lavigne.

CON 296 611 Mariana Leal - @lightesthand on twitter.

CON 297 612 Marie LeBlanc Flanagan // @omarieclaire.

CON 298 613 Jorge Ledezma.

CON 299 614 Elena Lee Gold (@elenaleegold).

CON 300 615 Lauren Lee McCarthy.

CON 301 616 Ahree Lee / www.ahreelee.com.

CON 302 617 Joey Lee | https://jk-lee.com.

CON 303 618 New Media Artists Louise Lessél & Jingyu Wen, @louiselessel @monica_jingyi.

CON 304 619 Golan Levin.

CON 305 620 Li, Yiqi Eyes-Experimental Clock (https://kiwili-portfolio.glitch.me/p3.html).

CON 306 621 LIA (www.liaworks.com).

CON 307 622 karlos g liberal (@patxangas).

CON 308 623 Daniel Lichtman.

CON 309 624 Jeff Lieberman + Bill Washabaugh.

CON 310 625 Wolf Lieser (DAM).

CON 311 626 Kenneth Lim, limzykenneth on Github.

CON 312 627 Adelle Lin, @adellelin, https://github.com/adellelin.

CON 313 628 Justin Lincoln.

CON 314 629 Liquido Preto, instagram.com/liquidopreto.

CON 315 630 @lisajamhoury.

CON 316 631 Connie Liu Twitter/IG@conliuart.

CON 317 632 Katie Liu (https://katieliu.art/).

CON 318 633 loackme.

CON 319 634 https://twitter.com/IOivm.

CON 320 635 Antimodular Research / Rafael Lozano-Hemmer's studio / lozano-hemmer.com.

CON 321 636 Kristin Lucas.

CON 322 637 LUMEN, @estudiolumen.

CON 323 638 MARIA MACIAK: MAMA, SOIL, AND SPIRIT mariamaciak.com.

CON 324 639 Organizado por Brian Mackern, Luisa Pereira y Daniel Argente.

CON 325 640 Nikki Makagiansar.

CON 326 641 makio135.

CON 327 642 Maya Man, mayaontheinter.net, @mayaontheinternet.

CON 328 643 Marcela Mancino // github: mardefronteira // ig: marcelamancino.

CON 329 644 manoloide.com.

CON 330 645 Vinicius Manrique.

CON 331 646 William Mapan – Generative Artist (tw: @williamapan).

CON 332 647 Gus Marcos.

CON 333 648 マリエちゃん.

CON 334 649 Srdjan Markovic.

CON 335 650 Ricard Marxer / @ricardmp.

CON 336 651 evelyn masso (@outofambit).

CON 337 652 https://github.com/matheplica.

CON 338 653 @MathMakesArt.

CON 339 654 @MattKaneArtist.

CON 340 655 Kyle McDonald @kcimc.

CON 341 656 Nick McIntyre.

CON 342 657 Joe McKay.

CON 343 658 Chandler McWilliams.

CON 344 659 Laleh Mehran.

CON 345 660 Jesal Mehta.

CON 346 661 Manu Mei-Singh, instagram handle: 01.manu.01.

CON 347 662 Jerome MERCIER - pizza-punk.com - Twitter @generativepunk.

CON 348 663 メリー@fnXzxnwcP6iZlUn.

CON 349 664 Guinevere Mesh (thedigitalguin.com / @thedigitalguin).

CON 350 665 Huw Messie - Textiles and New Media Artist - @huwmessie @huwythechew.

CON 351 666 Sandro Miccoli (@sandromiccoli/@code2pixels_).

CON 352 667 Andrea Misuraca / @instamisu.

CON 353 668 Congratulations Processing on your 20th Anniversary! Many Thanks to the Processing Community! Processing is a very helpful tool for me to explore my interests in photography and the Arts. I have been using Processing since 2010 starting with version 2. The improvements over the years have certainly helped expand the range of projects where I could apply Processing. I really enjoy writing code and Processing helps me satisfy my craving. I started coding in 1963 beginning with FORTRAN using punched card input. Thankfully we are much more productive today with the many tools we have like Processing. It helps that we can write code in Processing that runs on multiple hardware and software platforms. Andy Modla Software Engineer and 2D/3D Photographer Github: @ajavamind Blogs with some postings

related to Processing: https://andymodla3dvr.blogspot.com/ https://andymodlaphotography.blogspot.com/ https://rcastudio2videogame.blogspot.com/ Instagram: @andy_modla_photography @tekla3d.

CON 354 669 mole^3.

CON 355 670 Stig Møller Hansen (@Stixan · stigmollerhansen.dk).

CON 356 671 aarón montoya-moraga.

CON 357 672 Harvey Moon and Qianqian Ye.

CON 358 673 Harvey Moon.

CON 359 674 Eduardo Morais.

CON 360 675 Ben Moren.

CON 361 676 Morpholux - Jean-François Renaud.

CON 362 677 mr_samfield.

CON 363 678 Ardak Mukanova https://www.behance.net/ardak_mukanova https://www.instagram.com/ardakmukanova.

CON 364 679 Siddhartha Mukherjee, https://www.instagram.com/decodingkunst.

CON 365 680 Siddhartha Mukherjee, https://www.instagram.com/decodingkunst (TU Delft).

CON 366 681 Dónal Mulligan, @ donalmulligan.

CON 367 682 Jaime Munarriz.

CON 368 683 Musings with Code (Twitter: @ musingswithcode).

CON 369 684 name: Tatyana Mustakos - website: tademu.com - Instagram: @ poorly_documentedd.

CON 370 685 @nacho_cossio.

CON 371 686 @nagayama.

CON 372 687 Till Nagel (@tillnm) & Christopher Pietsch (@ chrispiecom).

CON 373 688 Agoston Nagy.

CON 374 689 Yota Nakamura (ohayota).

CON 375 690 남궁민상 (Minsang Namgoong) / KAIST 문화기술대학원 (Graduate School of Culture Technology, KAIST).

CON 376 691 Ram Narasimhan - github.com/Ram-N.

CON 377 692 Shefali Nayak [github @ shefalinayak].

CON 378 693 ねじおさん.

CON 379 694 NEORT.

CON 380 695 Nervous System @ nervous_system http://nervo.us.

CON 381 696 insta@newmiddleclass.

CON 382 697 nicolas_lebrun_@ twitter.com nclslbrn@github.com nicolas-lebrun.fr.

CON 383 698 Electronics Engineer && Data Scientist https://twitter.com/nicolasbaez.

CON 384 699 https://www.instagram.com/no.fill_/.

CON 385 700 Noite de Processing https://garoa.net.br/wiki/Noite_de_Processing.

CON 386 701 Widianto Nugroho, @ widiantonugroho (Instagram/GitHub).

CON 387 702 Kofi Oduro (Illestpreacha), Illestpreacha on all platforms and website: https://portfolio.illestpreacha.com/ links platforms website

CON 388 703 Old man but processing newbie.

CON 389 704 Krister Olsson • @ kristolsson (Twitter).

CON 390 705 Alex Olwal, www.olwal.com.

CON 391 706 Alex Olwal, Jon Moeller, Greg Priest-Dorman, Thad Starner, Ben Carroll, olwal.com/iobraid.

CON 392 707 Takafumi Oyama, Experience Engineer.

CON 393 708 Christian Oyarzún Roa / coyarzun@error404.cl / https://github.com/coyarzun/.

CON 394 709 Zeynep Özcan, @ ozeyna (instagram).

CON 395 710 @p1x3lboy.

CON 396 711 p1xelfool.

CON 397 712 p5.js website Japanese translation team.

CON 398 713 Dave Pagurek.

CON 399 714 Aamina Palmer, Artist-Designer, Instagram @dalaiami, amipalm.com.

CON 400 715 Priti Pandurangan. Instagram : priti.pg.

CON 401 716 Heracles Papatheodorou • @Arty2 • https://heracl.es.

CON 402 717 Evolutionary System to Design Type – Jéssica Parente, University of Coimbra, CISUC, DEI.

CON 403 718 Devi Parikh.

CON 404 719 Name: Jeongho Park, Title: Scan Sequencer Javascript, Gitbug Source Code: https://github.com/jeonghopark/scanseqjs, Twitter: https://twitter.com/jeonghopark.

CON 405 720 Name: Joo Park; ig: @ joopark.jpg.

CON 406 721 Robin Parmar.

CON 407 722 Adriano Parracciani aka Cyberparra.

CON 408 723 Kate Parsons, Assistant Professor at Pepperdine University, Co-Founder at FLOAT.

CON 409 724 Student self-portraits from the Computer Science for Designers & Artists course taught by Nikita Pashenkov at ArtCenter College of Design (2019 - 2021), from top left to bottom right: E Jun An, Ivy Li, Max Wright, Chanmee Park, Yuheng Xie, Fulya Akoz, Christine Choi, Miles Ogden, Devin Fung, Mengshu Liu, Michael Tu, Taiga Haruyama, Monica Zhang, Monica Hwang, Gyu Hyung Kang, Ariana Pacino, Jaewoo Ma, Marianne Wellman, Jun Seok Lee, Zachary Fua, David Pentland, Jacqlin Ha, Benjamin Beserra, Boyuan Shi, Steven Mao, Humberto Rivera, Adhi Wonowidjojo, Christine Wei, Robin Keum, Hanyu Bao, Siladityaa Sharma.

CON 410 725 @patternseeing.

CON 411 726 Name: Mou Peijing Title: Fukushima Offset Social Media: @melodrama_yu Website: https://www.melodymou.com/.

CON 412 727 Aaron Penne (@ aaronpenne).

CON 413 728 Federico Pepe @ fedpep.

CON 414 729 PERLINDRAW (Paulo Costa - NAVAX) - https://www.facebook.com/processingsp.

CON 415 730 Phoenix Perry.

CON 416 731 Niklas Peters.

CON 417 732 Kyle Phillips.

CON 418 733 Noah Travis Phillips (.com) / @th3n04h.

CON 419 734 my github: https://github.com/PhotonNikko.

CON 420 735 Gonzalo Piérola Azanza.

CON 421 736 Pitheorem [Pedram Sadeghbeyki].

CON 422 737 Matthew Plummer-Fernández.

CON 423 738 Stéphane Pogran, https://www.instagram.com/stephanepogran/.

CON 424 739 Niels NTG Poldervaart.

CON 425 740 Juan Carlos Ponce Campuzano, @ jcponcemath.

CON 426 741 On the importance of words, Panayota Pouliou @pitsypeach.

CON 427 742 H.Prakash (Blogger, Animator, Geek), moarpixels.tumblr.com, Twitter/Instagram:@ prakashph.

CON 428 743 Process – www.process.studio (Martin Grödl, Moritz Resl).

CON 429 744 Processing Buenos Aires - Argentina (2014-2015).

CON 430 745 Processing Paris.

CON 431 746 James Proctor @jprctr.

CON 432 747 George Profenza, @ orgicus.

CON 433 748 Martin Prout aka monkstone / @ monkstoneT.

CON 434 749 Iris Qu (instagram: iris_xiaoyu).

CON 435 750 Achilleas Rachoutis ig: achilleas_rach.

CON 436 751 Daniel Raedel, PsyD.

CON 437 752 Fernanda Ramos, visual artist, www.fernandaramos.com.

CON 438 753 Oz Ramos.

CON 439 754 @rand0mwalk; williamlockett.org.

CON 440 755 randomseed.cargo.site.

CON 441 756 Tiemen Rapati (@ rapatski) / United Visual Artists.

CON 442 757 Rasagy / @ data.n.coded.

CON 443 758 Fabin Rasheed | www.nurecas.com | twitter.com/fabinrasheed.

CON 444 759 @REAS.

CON 445 760 Andreas Refsgaard, https://twitter.com/AndreasRef, https://www.instagram.com/andreasref/.

CON 446 761 Chris Reilly.

CON 447 762 @reona396.

CON 448 763 Moritz Resl, Paul Sommersguter.

CON 449 764 N.Restelli.

CON 450 765 Everardo Reyes.

CON 451 766 rich.gg / Digital Architect.

CON 452 767 Matt Richard, Wings, @isocor.

CON 453 768 Roland Richter.

CON 454 769 Concept and research: Renée Ridgway, Data visualisations: Richard Vijgen.

CON 455 770 Chris Ried / Twitter: @ generatecoll.

CON 456 771 Frederik Riedel, https://riedel.wtf.

CON 457 772 “Anamika” - Rishi (@DenisovichPy).

CON 458 773 @robotandproud.

CON 459 774 /Cátia Roça.

CON 460 775 Tim Rodenbröker, timrodenbroeker.de.

CON 461 776 Nikita Rokotyan.

CON 462 777 Angela Rong, https://github.com/togekisse.

CON 463 778 h. scott roth, Twitter: hscottroth.

CON 464 779 A. Rothaug.

CON 465 780 Rebecca Ruige Xu, Sean Zhai.

CON 466 781 Lucrezia Russo / IG: @lucreziarusso.

CON 467 782 Soundbird (2008) by rux (Rui Pereira) @rux_twitts_here.

CON 468 783 RVig, Generative Artist, @rvig_art.

CON 469 784 Tom-Lucas Säger Twitter @t00may Instagram @tlsaeger.

CON 470 785 Ashna Sahir.

CON 471 786 Santino Santos (Instagram/Twitter handles = samoandrop123).

CON 472 787 @sarah_ridgley.

CON 473 788 Emi Sato, Designer, emi-sato.com, @emisato02.

CON 474 789 Takayuki Sato (@ takayukisato624).

CON 475 790 Lasse Scherffig.

CON 476 791 Mark Schifferli.

CON 477 792 Andreas Schlegel @sojamo, ControlP5.

CON 478 793 Andreas Schlegel @sojamo and Brian O’Reilly.

CON 479 794 Jim Schmitz.

CON 480 795 Johanna Schneider.

CON 481 796 Derrick Schultz (IG: @dvsmethid, Twitter: @dvsch).

CON 482 797 Marcel Schwittlick.

CON 483 798 eric socolofsky (@ericsoco).

CON 484 799 Chris Sears.

CON 485 800 phil-seaton.com/.

CON 486 801 Senbaku.

CON 487 802 Instagram - @ senseiwhocodes.

CON 488 803 James Jung-Hoon Seo (@lossless).

CON 489 804 Gerard Serra.

CON 490 805 Ayush Sharma, Engineer (github: @a-y-u-s-h, twitter: @taggosaurus).

CON 491 806 Jianan Shi, UCL student @interactive architecturelab, ins@ sigua2021.

CON 492 807 Yining Shi, https://twitter.com/yining_shi.

CON 493 808 Yujing Shi (https://yujingsss.github.io/yjs/).

CON 494 809 Munus Shih, Design/Coder, https://munusshih.cargo.site.

CON 495 810 Shervin Shirmohamadi, Undergraduate student Mechanical Engineering , Sbu , Iran / ( my instagram link )( https://www.instagram.com/shervin.shirmohamadi/).

CON 496 811 www.patshiu.com.

CON 497 812 Neel Shivdasani (@neel.shivdasani).

CON 498 813 Shubhang-sharma.

CON 499 814 Hisatomi Shuji (instagram : @shuji15.904).

CON 500 815 Takawo Shunsuke.

CON 501 816 Aaron Siegel, datadreamer.com.

CON 502 817 vladimir sierra https://www.instagram.com/plutovman.

CON 503 818 Carrie Sijia Wang, Artist/Designer/Educator, Instagram: carrie_re7l.

CON 504 819 Sinan from OpenProcessing.

CON 505 820 Kevin Siwoff.

CON 506 821 Julio Smitter / JSF.

CON 507 822 Sofia S.G. a.k.a. Sofia Rossi Torres a.k.a. sofurby.

CON 508 823 Name: Gaurav Sohani, Github: scorpiyogi, Twitter: scorpiyogi1.

CON 509 824 Lex Sokolin, Artist, https://twitter.com/UAesthete.

CON 510 825 Joan Soler-Adillon.

CON 511 826 KT Son @4praktice.

CON 512 827 Winnie Soon & Geoff Cox, Aesthetic Programming, www.aesthetic-programming.net.

CON 513 828 Marcelo Soria-Rodriguez, @msoriaro.

CON 514 829 So Sun , My Journey with P5js , instagram @sosunnyproject , youtube.com/sosunnyproject.

CON 515 830 @spacepolygon.

CON 516 831 https://www.instagram.com/spaghetticoder77/.

CON 517 832 Barry Spencer @ speculatype.

CON 518 833 @spoonrider.

CON 519 834 Starkeffect (Twitter: @ starkeffectart, IG: @ starkeffect).

CON 520 835 Amanda Stojanov.

CON 521 836 Sumit_Saini https://github.com/sumqwerty.

CON 522 837 Alida Sun all social: @alidasun linktr.ee/alidasun.

CON 523 838 Tim Sun.

CON 524 839 Maks Surguy (@ msurguy).

CON 525 840 Crosser, rewritten in P5.js and Play by the SWEAT Collaborative.

CON 526 841 Artist: Tuang T, Instagram: @TuangStudio.

CON 527 842 Daniele Tabellini @ fupete.

CON 528 843
tabreturn.com.

CON 529 844
Taiwan Contemporary Culture Lab (C-LAB).

CON 530 845
Muhammed Takriti. Entrepreneur, generative art coder.

CON 531 846
Jared S Tarbell.

CON 532 847
Izza Tariq, Software Engineer, LinkedIn (www.linkedin.com/in/izza-tariq-707181b2/) , GitHub (https://github.com/Izza11).

CON 533 848
@tasty_plots.

CON 534 849
Instagram: @tathagatparihar.

CON 535 850
Scott Tatsuta.

CON 536 851
The Coding Train.

CON 537 852
The funprogramming.org community + Abe Pazos.

CON 538 853
The Realität Team (MX).

CON 539 854
Richard The (he/him) Studio TheGreenEyl | Parsons School of Design.

CON 540 855
Various projects created by Studio TheGreenEyl where Processing or P5 were used in the process. www.thegreeneyl.com.

CON 541 856
Echo Theohar, @eskyet.

CON 542 857
Nic Thibodeaux (Twitter: @nicthibs, @ConwaysCorridor).

CON 543 858
jesse c thompson.

CON 544 859
Ryan Thompson - (Twitter) @RT_Artwork - www.ryanthompson.name.

CON 545 860
Chelsea Thompto - Assistant Professor of Digital Media Art at San José State University. https://github.com/cthompto.

CON 546 861
Nicolas Tilly.

CON 547 862
Woodcut Flow by Jason Ting (instagram.com/jzlabs).

CON 548 863
Tiny Tech Zines.

CON 549 864
架旗 透 (Kahata Toru) / Sci-Fi Writer/ @Hodler_SciFi.

CON 550 865
Nitcha Tothong, Interdisciplinary Artist and Designer, @nitchafame, https://www.nitcha.info/.

CON 551 866
@toxi.

CON 552 867
Anya Tran and Lily Zhou.

CON 553 868
Amy Traylor (Mom, Artist, Teacher, Coder).

CON 554 869
Lee Tusman.

CON 555 870
USC IML300&400.

CON 556 871
Niels V..

CON 557 872
Jakub Valtar.

CON 558 873
Vamoss - https://vamoss.com.br.

CON 559 874
Frederik Vanhoutte (@wblut).

CON 560 875
@variable_j.

CON 561 876
Roopa Vasudevan, media artist & researcher // @rouxpz.

CON 562 877
Iskra Velitchkova.

CON 563 878
Sergio Venancio.

CON 564 879
Marta Verde. Personal website: www.martaverde.net Instagram: @martaverdebaqueiro.

CON 565 880
Jeroen Verschuren - www.seads.nework.

CON 566 881
Jeroen Vesseur.

CON 567 882
José Vicente Araújo / DUNADIGITAL.com.

CON 568 883
Sai Ram Ved V (@sairamved).

CON 569 884
Santi Vilanova - Playmodes Studio.

CON 570 885
Alexandre Villares, André Burnier, Bernardo Fontes, Bárbara Castro, Carlos de Oliveira, Guilherme Vieira, Helena Sulzbeck, Monica Rizzolli, Paulo Costa, Rodrigo Junqueira, Victoria Pianca, Vinícius Padilha.

CON 571 886
Alexandre B A Villares https://abav.lugaralgum.com.

CON 572 887
Kawandeep Virdee, @whichlight, whichlight.com.

CON 573 888
vlak.xyz.

CON 574 889
Simon Wainwright.

CON 575 890
Name: Henry Haoyu Wang Title: artist/creative technologist Github: https://github.com/henrywang95 Instagram: henryiswhy.

CON 576 891
Doeke Wartena, www.doekewartena.nl.

CON 577 892
Marius Watz @mariuswatz.

CON 578 893
Mark Webster.

CON 579 894
Chris Welch - @chriswelch_72.

CON 580 895
Dave Whyte @beesandbombs.

CON 581 896
David Wicks @sansumbrella.

CON 582 897
@williamngan.

CON 583 898
Amelia Winger-Bearskin.

CON 584 899
Leisure Wolf.

CON 585 900
Mike Wong (@artixels).

CON 586 901
Professor Danne Woo / Queens College, CUNY / creativecode.dannewoo.com.

CON 587 902
Kirk Woolford ( https://b.bhaptic.net ).

CON 588 903
Kevin Workman - HappyCoding.io.

CON 589 904
Andrew J Wright | @concept_blend.

CON 590 905
Kuan-Ju Wu.

CON 591 906
Copyright: X|Y lab - Open Source project - development team: Stefano Adamo, Eugenio Battaglia, Danilo Di Cuia, Alessio Erioli, Nicolò Loprieno, Giulia Marzin, Piero Molino, Michele Pastore, Leonardo Romei, Matteo Stefanielli, Antonio Vergari. Year: 2014 (July 15-31) Link: https://www.co-de-it.com/open-source-neural-headset-development.html - code: https://bitbucket.org/ddicuia/openeeg/src/master/.

CON 592 907
Emily Xie, Generative Artist. @emilyxxie.

CON 593 908
@xrispy.

CON 594 909
Shuyu Xu.

CON 595 910
Masaki Yamabe, Data Visualization Designer, @masakick.

CON 596 911
ins:@yaogeeziyii.

CON 597 912
Yazid, Generative Artist, @Yazid (Twitter).

CON 598 913
Qianqian Ye.

CON 599 914
PCD Shanghai, organized by Qianqian Ye, Chang Liu, and Yuli Cai.

CON 600 915
Title: Processing Foundation 2020 Project “p5 for 50+”, and “50+ Coding Club Workshop series” Project Link: https://p5for50.plus | Project Lead: Inhwa Yeom, Seonghyeon Kim | Mentor: Qianqian Ye Advisor: Lauren McCarthy | Project Sponsor: Processing Foundation | Workshop Lead: Inhwa Yeom | Workshop Organizer & Planner: Sun Lee (Curator, Lee Kang Ha Arts Museum) | Workshop Teaching Assistant: Cholmin Kang | Workshop Sponsor: Asia Culture Center(ACC), Gwangju, South Korea.

CON 601 916
Title: Google Summer of Code 2020 Project “p5 for 50+ teaching” | Student Developer: Inhwa Yeom | Mentor: Qianqian Ye | Hosted by Processing Foundation | Funded by Google | Page Link: https://p5js.org/teach | GitHub Link: https://github.com/processing/p5.js/blob/main/contributor_docs/project_wrapups/inhwayeom_gsoc_2020.md.

CON 602 917
Tyler Yin.

CON 603 918
Logan K. Young.

CON 604 919
z1 studio (P5) and neoneo (Graphic Design).

CON 605 920
Ksen Zero (http://xnze.ro; IG: @xn_ze_ro) - all images from publicly-available sketches on https://openprocessing.org/class/70239.

CON 606 921
Wenran ZHAO is an artist who focuses on code art, 3D animation, painting and street art.

CON 607 922
Yufeng Zhao; once upon a time; github: yz3440; ig: hallucitalgia.

CON 608 923
Annie Zheng.

CON 609 924
Yayuan Joyce Zheng | Experience Designer | ins:@joyceiszyy.

CON 610 925
Lingyi Zhou.

CON 611 926
Name: Yiqing Zhou.

CON 612 927
Jan Zizka, Alex Olwal, Ramesh Raskar, olwal.com/specklesense.

CON 613 928
Michael Zoellner / #emnullfuenf.

CON 614 929
Mattia Zucchelli.

CON 615 930
Nolan Zurek.

CON 616 931
Anonymous.

Editors
Lauren Lee McCarthy
Casey Reas

Community
Coordinator
Qianqian Ye

Research and
Editorial Assistant
Nikki Makagiansar

Designers
New Information
Stefanie Tam
Inyeong Cho
Dave Yun

Proofreader
Sarah Dorhmann

Additional Copyediting
Johanna Hedva
Tess Thackara

Printer
Pristone, Singapore

Typeface
Suisse Int'l

Published
New York, NY
2022
The Processing
Foundation

ISBN (Regular Edition)
978-0-9998813-2-3

Executive Director
Dorothy R. Santos

Education Community
Director
Saber Khan

Grants & Finance
Manager
Toni Pizza

Project Leads
evelyn masso and
Qianqian Ye, p5.js
(Previously Lauren
Lee McCarthy,
Moira Turner)
Cassie Tarakaijan,
p5.js Editor
Jonathan Feinberg,
Processing.py
Ben Fry and Casey
Reas, Processing

| | |
|---|---|
| | Processing 3.0 Release |
| | Processing Libraries |
| | Processing People |
| | Processing and FLOSS |
| | Processing.py |
| | Processing for Pi Technical Notes |
| | The New Processing for Android |
| | New Processing.org |
| P5.JS | First p5.js Contributors Conference |
| | p5.js Community Statement |
| | p5.js Friendly Error System |
| | p5.js Accessibility Project |
| | Diversity with Code + Art |
| | p5.js Internationalization |
| | Second p5.js Contributors Conference |
| | p5.js Access Statement |
| | p5.js 1.0 Release |
| | p5.js x W3C |
| | p5.js Libraries |
| | p5.js Leadership Transition |
| PROCESSING COMMUNITY DAY | 2016 Community Survey |
| | Boston |
| | Los Angeles |
| | Worldwide |
| | Worldwide, Processing 20th Birthday |
| | Processing Community Day Organizer's Kit |
| SOFTWARE RELEASES | Processing |
| | p5.js |
| | p5.js Editor (v1) |
| | p5.js Editor (v2) |
| | p5.Sound |
| | Processing for Android |
| CONTRIBUTIONS | Contributions Index |
| | Contributions |